AKADEMIA OBRONY NARODOWEJ

WYDZIAŁ WOJSK LĄDOWYCH

KATEDRA SZTUKI OPERACYJNEJ

Do użytku służbowego

Egz.nr ... 2

MODELOWANIE OPERACJI WOJSK LĄDOWYCH

/MODEL - 2/

MODUŁ WALKI RADIOELEKTRONICZNEJ

Projekt technologiczny

WARSZAWA 1990

AKDEMIA OBRONY NARODOWEJ

WYDZIAŁ WOJSK LĄDOWYCH

KATEDRA SZTUKI OPERACYJNEJ

MODELOWANIE OPERACJI WOJSK LĄDOWYCH

(MODEL - 2)

MODUŁ WALKI RADIOELEKTRONICZNEJ

Projekt technologiczny

płk doc. dr hab. W. Brzostek
ppłk dr K. Sikorski
ppłk mgr P. Konca

WARSZAWA 1990

SPIS TREŚCI

1. MODYFIKACJA KONCEPCJI REALIZACJI MODELU OBEZWŁADNIANIA RADIOELEKTRONICZNEGO ŚRODKÓW I SYSTEMÓW DOWODZENIA PRZECIWNIKA

Przedmiotem odwzorowań modelu obezwładniania radioelektronicznego środków i systemów dowodzenia przeciwnika w/g przyjętej koncepcji rozwiązania tematu miał być armijny system obezwładniania radioelektronicznego.
Za miarę efektywności funkcjonowania tego systemu przyjęto stosunek ilości zakłóconych relacji łączności w sieciach dowodzenia nieprzyjaciela do ilości wszystkich czynnych relacji łączności w jednostce czasu (np. 1 godz. walki, doba bojowa itp.).
W wyniku dyskusji nad przedstawioną koncepcją w poprzednim etapie pracy oraz koniecznością zastosowania innej generacji techniki informatycznej (IBM AT a nie SIRIS-8), autorzy zdecydowali się na jego modyfikację. Modyfikacja dotyczy w zasadzie dwu spraw. Po pierwsze rozszerzenia modelu poprzez odwzorowanie w nim oddziaływania środków zakłóceń nieprzyjaciela na systemy łączności wojsk własnych, a zatem na przyjęcie modelu obezwładniania radioelektronicznego środków i systemów dowodzenia przeciwnika jako modelu dualnego.
Po drugie wprowadzenia dodatkowej funkcji oceny efektywności systemu obezwładniania definiowanej średnim czasem opóźnienia informacji zarówno w odniesieniu do poszczególnych ogniw dowodzenia jak i średniego czasu opóźnienia w całym rozpatrywanym systemie dowodzenia.
Ponadto wprowadzono szereg nowych rozwiązań ułatwiających samodzielną eksploatację modułu walki radioelektronicznej w procesie dydaktycznym akademii. W związku z powyższym założenia metodyczne algorytmu uległy odpowiedniej korekcie.

2. ALGORYTM

Program umożliwia realizację następujących funkcji:
a. tworzenie i aktualizację zbiorów danych stałych o węzłach łączności i środkach zakłóceń jednostek wojsk własnych i nieprzyjaciela;
b. zestawianie zbiorów danych zmiennych - sieci węzłów łączności i środków zakłóceń - z możliwością wykorzystania informacji ze zbiorów bazowych, utworzonych w punkcie a;
c. wykonanie obliczeń (symulacji) na zestawie danych zmiennych utworzonym w punkcie b.

2.1. Tworzenie i aktualizacja zbiorów danych stałych

Program umożliwia obsługę następujących zbiorów danych stałych:
- WEZLACZ.NIE - dane o węzłach łączności niebieskich;

- WEZLACZ.CZE - dane o węzłach łączności czerwonych;
- SRODZAK.NIE - dane o srodkach zakłóceń niebieskich;
- SRODZAK.CZE - dane o srodkach zakłóceń czerwonych;

Dla każdego z w/w zbiorów program umożliwia wykonanie następujących funkcji:

- założenie zbioru i wpisanie do niego informacji o obiektach;
- poprawianie zbioru:
  - dopisanie informacji o nowych obiektach;
  - kasowanie informacji o poszczególnych obiektach;
  - zmianę informacji o poszczególnych obiektach;
- wyświetlanie zawartości zbioru na monitorze;
- drukowanie zawartości zbioru na drukarce.

Zbiory danych o węzłach łączności: WEZLACZ.NIE i WEZLACZ.CZE mają identyczną budowę i składają się z rekordów typu WezLacz o strukturze:

- Przynal: string[10] - 10-znakowa nazwa jednostki, do której należy węzeł łączności;
- Nazwa: string[10] - 10-znakowa nazwa węzła łączności;
- X,Y: integer - współrzędne węzła łączności;
- LR: byte - ogólna liczba relacji węzła łączności;
- LRK: byte - liczba relacji KF węzła łączności;
- LRU: byte - liczba relacji UKF węzła łączności;
- LRR: byte - liczba relacji radioliniowych węzła łączności;
- DLM: integer - minimalna długość linii łączności dla danego węzła /km/;
- DMA: integer - maksymalna długość linii łączności dla danego węzła /km/;
- A: real - średnie natężenie ruchu dla danego węzła wyrażone w erlangach;
- PK: integer - moc promieniowania KF danego węzła /W/;
- PU: integer - moc promieniowania UKF danego węzła /W/;
- PR: integer - moc promieniowania radiolinii danego węzła /W/.

Założono, że w zbiorach WEZLACZ.NIE i WEZLACZ.CZE zapisane są dane o węzłach łączności poszczególnych jednostek. Dane o węzłach łączności każdej z jednostek są zapisane kolejno, jeden węzeł za drugim i nie powinny przeplatać się z węzłami łączności innych jednostek. W polu "Przynal" znajduje się nazwa jednostki do której należy opisywany węzeł. Jest to tekst o długości co najwyżej 10 znaków. Nazwa ta musi być identyczna dla wszystkich węzłów danej jednostki. W polu "Nazwa" znajduje się nazwa węzła łączności. Jest to również tekst o długości co najwyżej 10 znaków, służący do identyfikacji węzła. Współrzędne węzłów, w ramach danej jednostki podaje się przyjmując, że jednostka jest rozmieszczona w kwadracie o wierzchołkach (0, 100), (100, 100), (100, 0), (0, 0), przy czym front jednostki stanowi odcinek (0, 100), (100, 100) a tył odcinek (0, 0), (100, 0). Przyjmuje się tu jakieś ugrupowanie jednostki za standardowe i dla niego określa się współrzędne. Należy tu zwrócić uwagę, że przy przechodzeniu do konkretnego położenia jednostki (do danych zmiennych), można wykorzystać wprowadzone tu ugrupowanie standardowe jednostki i podać tylko czworokąt wypukły, w jakim znajduje się jednostka (podaje się tylko współrzędne czterech wierzchołków). Na tej podstawie program

oblicza położenia węzłów danej jednostki. Jest to oczywiście pewne uproszczenie, które jednak można wykorzystać np. gdy nie znamy rzeczywistych położeń węzłów łączności. Zawartości pozostałych pól nie wymagają komentarza.

Zbiory danych o środkach zakłóceń: SRODZAK.NIE i SRODZAK.CZE mają identyczną budowę i składają się z rekordów typu SrodZakl, o stukturze:
- Przynal: string[10] - 10-znakowa nazwa jednostki, do której należy dany środek zakłóceń;
- Nazwa: string[10] - 10-znakowa nazwa środka zakłóceń;
- X,Y: integer - współrzędne środka zakłóceń /km/;
- P: integer - moc środka zakłóceń /W/;
- CzR: real - czas reakcji środka zakłóceń /sek/;
- Zakres: byte - zakres zakłócany /1-UKF, 2-KF, 3-R/lin/.

Zasady utrzymywania informacji w tych zbiorach są podobne do zasad utrzymywania informacji w zbiorach danych o węzłach łączności. Dane o środkach zakłóceń każdej z jednostek powinny być zapisane kolejno, jeden środek za drugim i nie powinny przeplatać się ze środkami zakłóceń innych jednostek. Sposób wykorzystania pól "Przynal", "Nazwa" oraz współrzędnych "X" i "Y" jest identyczny jak dla węzłów łączności. To samo dotyczy przeliczania współrzędnych, przy przechodzeniu do danych zmiennych. Zawartości pozostałych pól nie wymagają komentarza.

2.2 Zestawianie zbiorów danych zmiennych.

Do wykonania obliczeń (eksperymentu symulacyjnego), informację o węzłach łączności i środkach zakłóceń zestawia się w dwu zbiorach WEZLACZ.OBL i SRODZAK.OBL. Zbiór WEZLACZ.OBL zawiera dane o węzłach łączności i składa się z rekordów identycznych jak zbiory WEZLACZ.NIE i WEZLACZ.CZE. Zbiór SRODZAK.OBL zawiera dane o środkach zakłóceń i składa się z rekordów identycznych jak zbiory SRODZAK.NIE i SRODZAK.CZE.
W odniesieniu do zbiorów danych zmiennych program umożliwia:
- selektywne przepisywanie informacji ze zbiorów danych bazowych (niebieskich lub czerwonych);
- wpisywanie (dopisywanie) z klawiatury informacji o nowych obiektach;
- kasowanie informacji o poszczególnych obiektach;
- zmianę informacji o poszczególnych obiektach;
- wyświetlanie na monitorze informacji o obiektach zapisanych w zbiorze;
- drukowanie na drukarce informacji informacji o obiektach zapisanych w zbiorze.

Do opisania położenia obiektów wykorzystywane są współrzędne. Należy tu zauważyć, że nie są to współrzędne topograficzne ani geograficzne, lecz zwykłe współrzędne prostokątne, w układzie wybranym przez użytkownika. Wielkość współrzędnych (zarówno "x" jak i "y") musi zawierać się w przedziale liczbowym (-999, 999).
Możliwe są dwa sposoby przepisywania informacji ze zbiorów danych bazowych do zbiorów danych zmiennych:
a. bez wprowadzania zmian w danych o poszczególnych obiektach;

b. z wprowadzaniem zmian.

W przypadku gdy nie wprowadza się zmian w danych o obiektach, dane są przepisywane ze zbiorów danych bazowych do zbiorów danych zmiennych z uaktualnieniem tylko współrzędnych. W tym wariancie podaje się tylko położenie (współrzędne czterech punktów) jednostki, której obiekty mają być wprowadzone. Na tej podstawie program oblicza położenie obiektów danej jednostki, posługując się wzorami:

$$Xf = Xfl + (x / 100) * (Xfp - Xfl)$$
$$Yf = Yfl + (x / 100) * (Yfp - Yfl)$$
$$Xt = Xtl + (x / 100) * (Xtp - Xtl)$$
$$Yt = Ytl + (x / 100) * (Ytp - Ytl)$$
$$X = Xt + (y / 100) * (Xf - Xt)$$
$$Y = Yt + (y / 100) * (Yf - Yt)$$

gdzie:

x,y - współrzędne "standardowe" obiektu (z danych bazowych);
Xfl,Yfl - współrzędne lewego skrzydła frontu jednostki;
Xfp,Yfp - współrzędne prawego skrzydła frontu jednostki;
Xtl,Ytl - współrzędne lewego skrzydła tyłu jednostki;
Xtp,Ytp - współrzędne prawego skrzydła tyłu jednostki;
Xf,Yf,Xt,Yt - wielkości pomocnicze;
X,Y - nowe, obliczone współrzędne obiektu.

W przypadku wprowadzania danych ze zbiorów bazowych z wprowadzaniem zmian użytkownik powinien sam uaktualnić współrzędne każdego obiektu.

2.3 Obliczenia

Podstawą eksperymentu są dystrybuanty (zał. 1), określone na podstawie danych empirycznych, obejmujących próbkę zgłoszeń, przechwyconych przez jednostki rozpoznania radioelektronicznego, w czasie trwania ćwiczenia "WINTEX/CIMEX-83" (ppłk dr inż. Waldemar Brzostek - Armijny system obezwładniania radioelektronicznego środków i systemów dowodzenia przeciwnika, rozprawa hab.).
W czasie obliczeń wykorzystywany jest bufor zgłoszeń aktualnych, znajdujący się w pamięci i zawierający dane o zgłoszeniach niezakończonych do danej chwili:

- czas nadejścia;
- czas trwania;
- rodzaj emisji;
- zakres;
- numer węzła łączności (źródła);
- numer stacji zakłócającej dane zgłoszenie (lub zero).

Dane o kolejnych zgłoszeniach są zapisywane w zbiorze ZGLOSZEN.OBL do wykorzystania przy tworzeniu wyciągów dla grup węzłów:

- czas nadejścia;
- czas trwania;
- numer węzła łączności (źródła);
- numer stacji zakłócającej (lub zero);
- rodzaj emisji;
- zakres;
- powód niezakłócenia (o ile nie zostało zakłócone).

Obliczenia (eksperymant symulacyjny) są prowadzone zgodnie z algorytmem podanym w załączniku 2.
Pierwszą czynnością jest wprowadzenie danych o węzłach łączności, opisanych w zbiorze WEZLACZ.OBL i środkach zakłóceń opisanych w zbiorze SRODZAK.OBL oraz zestawienie globalnej dystrybuanty i obliczenie globalnego średniego natężenia ruchu (PT ,AS w bloku 1 algorytmu).
Następnie program generuje kolejno zgłoszenia i sprawdza możliwość ich zakłócenia.
Dla każdego zgłoszenia są generowane następujące wielkości:

a. Czas nadejścia zgłoszenia (blok 3). Generowany jest na podstawie globalnego średniego natężenia ruchu (AS).
b. Źródło zgłoszenia (blok 4). Na podstawie globalnej dystrybuanty zestawionej w czasie wprowadzania danych, określa się numer węzła. Następnie na podstawie ilości wolnych relacji w poszczególnych zakresach (UKF, KF, R/1), generuje się zakres zakładając, że prawdopodobieństwo wylosowania zakresu jest proporcjonalne do liczby wolnych relacji danego zakresu.
c. Rodzaj emisji (blok 5-7). W zależności od zakresu generuje się:
   - dla zakresu KF rodzaj emisji 1, 2 lub 3 przyjmując, że prawdopodobieństwo wystąpienia danego rodzaju emisji jest proporcjonalne do liczby jego wystąpień w dystrybuantach empirycznych poszczególnych rodzajów emisji;
   - dla zakresu UKF rodzaj emisji 1 lub 3 przy założeniach jak wyżej;
   - dla R/1 przyjmuje się rodzaj emisji 1.
d. Czas trwania zgłoszenia (blok 8). Generuje się zgodnie z dystrybuantą empiryczną dla wygenerowanego wcześniej rodzaju emisji. Na podstawie czasu nadejścia zgłoszenia i czasu trwania zgłoszenia oblicza się czas zakończenia zgłoszenia.

Po wygenerowaniu w/w wielkości, kasuje się w buforze zgłoszeń aktualnych, dane o zgłoszeniach, których czas zakończenia upłynął przed czasem nadejścia bieżącego zgłoszenia (blok 9).
Następnie (blok 10) tworzy się tablicę węzłów zakłócanych WezZak[i], wpisując do i-tej pozycji numer węzła, który zakłóca i-ta stacja zakłóceń lub zero gdy i-ta stacja nie zakłóca (jest wolna)..
Potem (blok 11) do bufora zgłoszeń aktualnych dopisuje się dane o zgłoszeniu bieżącym:

- czas nadejścia zgłoszenia;
- czas trwania zgłoszenia;
- rodzaj emisji;
- zakres;
- numer węzła łączności.

Następną czynnością (bloki 12-22) jest określenie możliwości zakłócenia danego zgłoszenia. W tym celu przeglądane są środki zakłóceń poczynając od położonego najbliżej danego węzła i dalej kolejno aż do położonego najdalej lub do znalezienia środka, który może zakłócać dane zgłoszenie. Zakłada się że środek zakłócający węzeł o niższym priorytecie może zostać "przecelowany" na zakłócanie węzła o wyższym priorytecie, natomiast środek zajęty zakłócaniem węzła o wyższym priorytecie nie może być użyty do zakłócania węzła o niższym priorytecie.

Oprócz tego sprawdzane są jeszcze warunki:
- czy środek może zakłócać zakres, w którym wystąpiło bieżące zgłoszenie (blok 14);
- czy środek ma wystarczająco krótki czas reakcji do zakłócenia bieżącego zgłoszenia tzn. czy czas trwania zgłoszenia nie jest mniejszy od czasu reakcji danego środka (blok 15);
- czy środek ma wystarczającą moc do zakłócenia danego zgłoszenia (blok 17).

Sprawdzenia czy środek ma wystarczającą moc do zakłócenia danej relacji dokonuje się w zależności od zakresu zgłoszenia.
Dla radiolinii przyjmuje się że zgłoszenie zostanie zakłócone niezależnie od mocy środka zakłócającego.
Dla zakresów UKF i KF generowana jest odległość Rs stacji nadającej od odbierającej, jako liczba losowa z przedziału (DLM, DMA), przy czym zakłada się rozkład równomierny. Następnie dla tak wygenerowanej odległości oblicza się współczynnik zakłóceń:

$$Kzz = \frac{Rs\sqrt{Psz}}{Rz\sqrt{P}}$$

gdzie:
Rs - wygenerowana odległość stacji nadającej od odbierającej;
Rz - odległość stacji odbierającej od stacji zakłóceń;
Psz - moc stacji zakłóceń;
P - moc stacji nadającej.

Przyjmuje się dla rodzaju emisji 1 i 3, że moc stacji zakłóceń jest wystarczająca gdy Kzz ≥ 1 a dla rodzaju emisji 2 gdy Kzz ≥ 4.

Jeśli w wyniku opisanego postępowania okaże się że zgłoszenie może być zakłócone, to numer węzła danego zgłoszenia zostaje zapamiętany w WezZakl[SZak], zaś czas zakończenia transmisji danego zgłoszenia w ZajSZ[SZak] (blok 20).

Jeśli zgłoszenie nie może być zakłócone to jako powód przyjmuje się (bloki 21-28):
- warunki energetyczne (za mała moc środków zakłóceń), jeśli był choć jeden środek, który spełniał wszystkie warunki do zakłócenia zgłoszenia (możliwość zakłocania zakresu zgłoszenia, zajętość, czas reakcji) za wyjątkiem warunków energetycznych;
- czas reakcji, jeśli nie zachodzi poprzedni przypadek i był choć jeden środek zakłóceń wolny lub zakłócający węzeł o niższym priorytecie, mogący zakłócać zakres danego zgłoszenia a nie mogący zakłócić zgłoszenia ze względu na zbyt długi czas reakcji;
- zajętość systemu, jeśli nie zachodzą poprzednie przypadki.

Informacje o zgłoszeniu są zapisywane w zbiorze ZGLOSZEN.OBL (blok 30).
Wyniki wyświetlane są na bieżąco podczas trwania eksperymentu. Po zakończeniu eksperymentu ogólna informacja o wynikach znajduje się na ekranie monitora.

Program umożliwia tworzenie wyciągów dla wybranych grup węzłów. wyciągi te są tworzone na podstawie danych o zgłoszeniach zapisanych w zbiorze ZGLOSZEN.OBL w trakcie eksperymentu.

3. PROGRAM

Program został napisany w języku Pascal (wersja Turbo Pascal 5). Dla uproszczenia uruchamiania został on podzielony na 9 modułów.

Moduł Podst zawiera deklaracje pewnych stałych, tekstów i procedur wykorzystywanych w różnych miejscach programu. Stałe dotyczą kodów ASCII klawiszy sterowania kursorem. Teksty są dyrektywami dla użytkownika, dotyczącymi wyboru wariantu. Procedury realizują następujące funkcje:

- Beep - sygnał dźwiękowy;
- Maska - czyści ekran i rysuje na nim ramkę z napisem KSO AGG;
- WypDyr - wyprowadzanie dyrektyw (tekstów) w wierszach 22 i 23 ekranu;
- Wybor - sterowanie wyborem użytkownika z przedstawionej listy propozycji.

Moduł WRE1Decl zawiera deklaracje stałych, typów i zmiennych globalnych (dla całego programu).

Moduł WRE2ZakB zawiera dwie procedury główne:

- DopWL - zapisywanie danych o węzłach łączności do zbioru danych bazowych;
- DopSZ - zapisywanie danych o środkach zakłóceń do zbioru danych bazowych.

Oprócz tego zawiera procedury wykorzystywane w w/w procedurach głównych oraz w procedurach innych modułów:

- DrukWWL - drukowanie nazwy pola o podanym numerze, rekordu typu WezLacz;
- DrukWarWL - drukowanie wartości pola o podanym numerze, rekordu typu WezLacz;
- CzytNWL - czytanie wartości pola o podanym numerze, rekordu typu WezLacz;
- CzytajWL - czytanie danych o węźle łączności;
- DrukWSZ - drukowanie nazwy pola o podanym numerze, rekordu typu SrodZakl;
- DrukWarSZ - drukowanie wartości pola o podanym numerze, rekordu typu SrodZakl;
- CzytWSZ - czytanie wartości pola o podanym numerze, rekordu typu SrodZakl;
- CzytajSZ - czytanie danych o środku zakłóceń.

Moduł WRE3AktB zawiera dwie procedury główne:

- PopWL - poprawianie danych o węzłach łączności w zbiorach bazowych;
- PopSZ - poprawianie danych o środkach zakłóceń w zbiorach bazowych.

Oprócz tego zawiera procedury wykorzystywane przez te procedury główne oraz przez procedury modułu WRE5CzyD:
- WEdWezel - wybór sposobu poprawiania danych o węźle;
- EdycjaWezlow - sterowanie edycją węzłów bieżącej jednostki;
- WPopWL - wybór sposobu poprawiania danych o węzłach danej jednostki;
- WEdSrodZakl - wybór sposobu poprawiania danych o środku zakłóceń;
- EdycjaSrodZakl - sterowanie edycją środków zakłóceń danej jednostki;
- WPopSZ - wybór sposobu poprawiania danych o środkach zakłóceń danej jednostki.

Poza tym moduł zawiera pewne lokalne procedury pomocnicze.

Moduł WRE4WypB zawiera cztery procedury główne:
- WyswWL (DrukWL) - wyprowadza na ekran (drukarkę) dane o węzłach łączności, znajdujące się w zbiorach danych bazowych lub w zbiorach danych zmiennych, zależnie od wartości zmiennej "NazwaZb";
- WyswSZ (DrukSZ) - wyprowadza na ekran (drukarkę) dane o środkach zakłóceń, znajdujące się w zbiorach danych bazowych lub w zbiorach danych zmiennych, zależnie od wartości zmiennej "NazwaZb".

Oprócz tego moduł zawiera lokalne procedury pomocnicze.

Moduł WRE5CzyD zawiera dwie procedury główne:
- WprowWezLacz - wprowadza dane o węzłach łączności do zbioru danych zmiennych, ze zbioru danych bazowych lub z klawiatury;
- WprowSrodZakl - wprowadza dane o środkach zakłóceń do zbioru danych zmiennych, ze zbioru danych bazowych lub z klawiatury.

Oprócz tego moduł zawiera lokalne procedury pomocnicze.

Moduł WRE6Obl zawiera procedurę główną Obliczenia oraz lokalne procedury pomocnicze. Procedura Obliczenia realizuje eksperyment symulacyjny wyświetlając na bieżąco wyniki na ekranie i zapisując dane o przebiegu eksperymentu do zbioru ZGLOSZEN.OBL.

Moduł WRE7WypW zawiera procedurę główną WyswWez i lokalne procedury pomocnicze. Procedura główna umożliwia zestawianie na podstawie zbioru ZGLOSZEN.OBL i wyświetlanie wyników eksperymentu dla wybranej grupy węzłów.

Moduł WRE stanowi część sterującą programu. Umożliwia użytkownikowi wybór funkcji do wykonania i zapewnia realizację tej funkcji przez wywołanie odpowiednich procedur z innych modułów.

4. INSTRUKCJA EKSPLOATACJI

Eksploatację programu można podzielić na trzy etapy:

1. tworzenie i aktualizacja zbiorów stałych o węzłach łączności i środkach zakłóceń czerwonych i niebieskich (zbiory WEZLACZ.NIE, WEZLACZ.CZE, SRODZAK.NIE, SRODZAK.CZE);
2. zestawianie zbiorów danych zmiennych o węzłach łączności (WEZLACZ.OBL) i środkach zakłóceń (SRODZAK.OBL), przy czym można wykorzystywać dane ze zbiorów stałych zestawione w etapie 1. a także wprowadzać z klawiatury dane o nowych obiektach (węzłach łączności lub środkach zakłóceń);
3. wykonanie eksperymentu na podstawie danych zmiennych, zestawionych w etapie 2.

Sterowanie pracą programu polega na wybieraniu przez użytkownika odpowiednich pozycji z ofert przedstawianych przez program, w postaci masek przedstawionych w załączniku 4.
Po uruchomieniu program przedstawia użytkownikowi maskę 1. Jeśli wszystko jest gotowe do eksperymentu tzn. są już właściwie zestawione zbiory danych zmiennych, można wybrać opcję 1 - Obliczenia i program bez dalszych pytań zacznie wykonywać eksperyment. W przypadku wybrania opcji 4 - Zakończenie działania, program zakończy działanie (nastąpi powrót do os-u lub Nortona).
W przypadku wybrania opcji 2 lub 3 program spowoduje uściślenie wymagań użytkownika przez rozbudowanie maski 1 do maski 2. Po wybraniu przez użytkownika czynności (Dopisywanie, Poprawianie, Wyświetlanie, Drukowanie), maska 2 zostanie rozbudowana do maski 3 dla określenia obiektów, w stosunku do których dana czynność ma być wykonana (Węzły łączności, Środki zakłóceń).
W przypadku, gdy wybrane było: poprawianie, wyświetlanie (na ekranie) lub drukowanie (na drukarce) danych zmiennych, program przechodzi do wykonania żądanej czynności bez dalszych pytań.
Jeśli zostało wybrane dopisywanie dla danych zmiennych lub operacje na danych stałych, program rozbudowuje maskę 3 do maski 4 dla ustalenia której strony (NIEBIESKICH czy CZERWONYCH) operacja dotyczy, i dopiero wtedy przejdzie do wykonania żądanej czynności.

4.1. Tworzenie i aktualizacja zbiorów stałych

W przypadku wybrania na masce 1 opcji: Zestawianie danych stałych, program wyświetla jeszcze maski 2, 3 i 4 dla sprecyzowania wymagań użytkownika.

W przypadku wyświetlania lub drukowania, po dokonaniu wyboru na masce 4, program wyświetli na ekranie lub wydrukuje na drukarce dane, w układzie pokazanym na maskach 13 (węzły łączności) lub 14 (środki zakłóceń).

W przypadku dopisywania, program zacznie wyświetlać kolejne wiersze maski 5 (węzły łączności) lub 6 (środki zakłóceń), czekając na wprowadzenie przez użytkownika odpowiednich wartości.

Na zakończenie danych o obiekcie, program zadaje dwa pytania:
- ZAPISAĆ DO ZBIORU (T/N)? - jeśli wprowadzone zostanie T to program zapisze dane o wprowadzanym obiekcie do zbioru, jeśli N to nie zapisze;
- CZY WPROWADZASZ NASTĘPNY ... (T/N)? - jeśli wprowadzone zostanie T to program rozpocznie od nowa tę maskę, żądając wprowadzenia danych o następnym obiekcie, jeśli wprowadzone zostanie N to nastąpi zakończenie wprowadzania i powrót do maski 1.

Dopisywanie obiektów do zbiorów danych stałych odbywa się niezależnie od tego, czy zbiór istniał przed rozpoczęciem operacji czy nie. Jeżeli nie istniał, to jest zakładany i jest do niego wpisywana informacja o obiektach, jeżeli istniał, to wprowadzana informacja jest dopisywana na końcu zbioru.

Poprawianie informacji o obiektach może być realizowane tylko wtedy, gdy jest już zbiór z danymi stałymi o wybranych obiektach. Poprawianie odbywa się najpierw na szczeblach jednostek (maska 7 dla węzłów łączności i 10 dla środków zakłóceń). Proponowane funkcje:
- Zachowanie "obiektów" danej jednostki;
- Skasowanie "obiektów" danej jednostki;
- Zachowanie "obiektów" reszty jednostek;
- Skasowanie "obiektów" reszty jednostek;

nie wymagają komentarzy. Należy tylko zauważyć, że po pierwszych dwu, program przechodzi do następnej jednostki (o ile taka istnieje w zbiorze) a w przypadku dwu ostatnich, program kończy działanie i przechodzi do maski 1.
Wybór funkcji:
Poprawianie "obiektów" danej jednostki
powoduje wyświetlenie maski 8 lub 11 (zależnie od typu obiektów) kolejno dla każdego obiektu jednostki.
Funkcje ZACHOWAĆ i SKASOWAĆ dotyczą tylko wyświetlanego obiektu.
Funkcja DOPISAĆ PRZED powoduje przejście do masek 5 lub 6 (zależnie od typu obiektu), i dopisanie wprowadzonego z klawiatury obiektu przed obiektem wyświetlanym.
Funkcja ZMIENIAĆ powoduje przejście do masek 8 lub 11. Na tych maskach dokonuje się poprawek odpowiednich pól aż do wprowadzenia T dla ostatniego pytania: CZY ZAKOŃCZYŁEŚ EDYCJĘ "OBIEKTU" (T/N)?. Program zapisuje wówczas poprawione dane o obiekcie i wraca do maski 7 lub 11, jeśli w danej jednostce jest jeszcze jakiś obiekt albo do masek 6 lub 10 gdy nie ma, albo do maski 1 gdy skończył się zbiór.

4.2. Zestawianie danych zmiennych

W przypadku wybrania na masce 2 funkcji: Poprawianie, Wyświetlanie lub Drukowanie, program uściśla tylko czego to ma dotyczyć (maska 3) i przechodzi do realizacji danej funkcji. Realizacja tych funkcji nie różni się od ich realizacji dla zbiorów danych stałych (nie ma tu tylko ustalania, której strony ma to dotyczyć -

CZERWONYCH czy NIEBIESKICH).
W przypadku wybrania na masce 2 funkcji: Dopisywanie, program pyta jeszcze jakich obiektów i której strony ma to dotyczyć (maski 3 i 4). Następnie przedstawia na masce 15 wykaz jednostek odpowiedniego zbioru stałego, w celu wybrania jednostek, których obiekty będą dopisane do zbioru danych zmiennych. Po wybraniu przez użytkownika jednostkido przepisania, program pyta CZY CHCESZ POPRZWIAĆ (T/N) ? (maska 16).
Jeżeli użytkownik chce zmienić co najwyżej nazwę jednostki i spowodować przeliczenie "standardowych" współrzednych obiektów jednostki do nowego położenia - powinien odpowiedzieć N. Wówczas program zapyta go o nową nazwę i położenie jednostki - maska 17 po czym przepisze obiekty tej jednostki do zbioru danych zmiennych i przejdzie do maski 15.
Jeżeli użytkownik chce poprawiać dane o obiektach wybranej jednostki, powinien na pytanie: CZY CHCESZ POPRAWIAĆ (T/N) ? odpowiedzieć T. Wówczas program będzie przedstawiał kolejne obiekty wybranej jednostki na maskach 9 lub 12. Postępowanie jest wówczas analogiczne, jak przy poprawianiu obiektów w zbiorach danych stałych. Po wyczerpaniu obiektów danej jednostki następuje powrót do maski 15.
Wyjście z maski 15 (zakończenie wybierania obiektów ze zbioru danych stałych) następuje przez wybranie opcji KONIEC WYBIERANIA Z BAZY. Następuje wówczas przejście do maski 10.
Jeżeli użytkownik będąc na masce 18 nie chce wprowadzać danych o dodatkowych obiektach z klawiatury, powinien nacisnąć N. Nastąpi wówczas przejście do maski 1.
Jeżeli użytkownik będąc na masce 18 chce wprowadzać dane o dodatkowych obiektach z klawiatury, powinien nacisnąć T. Nastąpi wówczas przejście do maski 5 lub 6. Postępowanie jest wówczas analogiczne jak przy dopisywaniu obiektów do danych stałych.

4.3. Obliczenia

Po wybraniu na masce 1 opcji "Obliczenia", program rozpocznie obliczenia, wyświetlając na bieżąco wyniki, w postaci pokazanej w załączniku 5, maska 1.
Podawana jest ogólna liczba zgłoszeń z rozbiciem na zakresy oraz liczba zgłoszeń zakłóconych i nie zakłóconych ze względu na: czas reakcji systemu, zajętość systemu oraz warunki energetyczne (moc), również z rozbiciem na zakresy. Efektywność zakłóceń dla poszczególnych zakresów i ogólna, są liczone jako stosunek liczby zgłoszeń zakłóconych do ogólnej liczby zgłoszeń. Poza tym podawany jest czas i liczba zgłoszeń w toku.
Po zakończeniu obliczeń i obejrzeniu wyników przez użytkownika (ew. wydrukowanie przez naciśnięcie klawiszy Shift-PrtScr), naciśnięcie dowolnego klawisza powoduje pytanie: CZY WYBIERASZ GRUPĘ WĘZŁÓW (T/N) ?
Jeżeli wprowadzi się N to program zakończy działanie i przejdzie do maski 1.
Jeżeli wprowadzi się T to program umożliwi wybór grupy węzłów

wyświetlając maskę 2 (zał. 5). Po wybraniu grupy węzłów należy wybrać opcję: KONIEC WYBIERANIA WĘZŁÓW. Program wyświetli wówczas wyniki eksperymentu dla wybranej grupy węzłów (maska 3, zał. 5). Dla każdego węzła podane zostaną następujące dane:

- liczba zgłoszeń i ich sumaryczny czas;
- liczba zgłoszeń zakłóconych;
- liczba zgłoszeń niezakłóconych ze względu na: czas reakcji (CzR), warunki energetyczne (War) i zajętość systemu (ZajS);
- efektywność zakłóceń jako procent zgłoszeń zakłóconych (%Zgl) oraz procent zakłóconego czasu zgłoszeń (%Czasu);
- średnie opóźnienie liczone w sekundach.

Wszystkie te wielkości są podawane dla poszczególnych zakresów i sumarycznie.

Można wydrukować wyniki na drukarce naciskając klawisze Shift-PrtScr. Po naciśnięciu dowolnego klawisza program przejdzie do wyświetlania kolejnej maski typu 3, itd. aż do wyczerpania masek tego typu (zależy to od liczebności wybranej grupy węzłów). Potem program wyświetli maskę 4 (zał. 5) zawierającą sumaryczne wyniki zakłóceń dla wybranej grupy węzłów (są to te same wielkości które były podawane dla poszczególnych węzłów wybranej grupy tylko dotyczą całej grupy). Tu również można wydrukować wyniki na drukarce przez naciśnięcie klawiszy Shift-PrtScr. Po naciśnięciu dowolnego klawisza na masce 4 (zał. 5), program ponowi pytanie:
CZY WYBIERASZ GRUPĘ WĘZŁÓW (T/N) ?
Można wówczas zakończyć działanie (klawisz N) lub przejść do wybierania kolejnej grupy węzłów (klawisz T).

Załącznik 1

DYSTRYBUANTY EMPIRYCZNE

Dystrybuanta E1			Dystrybuanta E2			Dystrybuanta E3		
Czas	Czest.	Dystr.	Czas	Czest.	Dystr.	Czas	Czest.	Dystr.
0	0	0	0	0	0	0	0	0
..-10"	168	168	..-10"	976	976	..-10"	1428	1428
10"-20"	159	327	10"-20"	131	1107	10"-20"	514	1942
20"-30"	165	492	20"-30"	46	1153	20"-30"	318	2260
30"-40"	98	590	30"-40"	12	1165	30"-40"	226	2486
40"-50"	64	654	40"-50"	6	1171	40"-50"	189	2675
50"-60"	71	725	50"-60"	8	1179	50"-60"	168	2843
1'-2'	237	962	1'-2'	15	1194	1'-2'	539	3382
2'-3'	115	1077	2'-3'	3	1197	2'-3'	246	3628
3'-4'	96	1173	3'-4'	3	1200	3'-4'	216	3844
4'-5'	90	1263	4'-5'	1	1201	4'-5'	134	3978
5'-..	659	1922	5'-..	1	1202	5'-..	754	4732

ALGORYTM OBLICZEŃ

Załącznik 2

Początek obliczeń

1. Czytanie danych.

Obliczenie: $PT_i = \sum_{k=1}^{i} A_k$

dla $i = 1, \ldots, n_w$

gdzie: n_w - liczba węzłów łączności,

A_k - średnie natężenie ruchu k-tego węzła łączności

Obliczenie: $AS = \sum_{k=1}^{n_w} A_k$

2. Czas := 0
Lzg := 1

1

3. Generowanie czasu nadejścia zgłoszenia o numerze Lzg:

$$Czas := Czas + \frac{-\ln(Rand)}{AS}$$

4. Generowanie węzła łączności zgłoszenia:

$$NrWez := \min_{1 \le i \le n_w} (Rand * AS < PT_i)$$

5. Generowanie zakresu zgłoszenia (KF, UKF, R/Lin):

Zakres:

UKF gdy $z < U$

KF gdy $U \le z < U+K$

R/Lin gdy $z \ge U+K$

gdzie:

$z = Rand * (U+K+R)$

U, K, R - liczba wolnych kanałów UKF, KF i radiolinii danego węzła

6. $U+K+R > 0$ — N

T

7. Generowanie rodzaju emisji:

- w przypadku zakresu KF:

$$IRE := \max_{1 \le i \le 3} \left(Rand < \frac{\sum_{k=1}^{i} D_k^{12}}{\sum_{k=1}^{3} D_k^{12}} \right)$$

- w przypadku zakresu UKF:

$$IRE := \begin{cases} 1 \text{ gdy } Rand < \dfrac{D_1^{12}}{D_1^{12} + D_3^{12}} \\ 3 \text{ w przeciwnym razie} \end{cases}$$

- w przypadku radiolinii:

IRE := 1

gdzie: D_k^{12} - liczba zgłoszeń o rodzaju emisji k (na ogólną liczbę zgłoszeń $\sum_{k=1}^{3} D_k^{12}$) z dystrybuanty eksperymentalnej.

8. Generowanie czasu trwania zgłoszenia:

$$CzasTr := T_{i-1} + Rand * (T_i - T_{i-1})$$

gdzie:

$$i = \max_k \left(Rand \le \frac{D_{IRE}^k}{D_{IRE}^{12}} \right)$$

T_i - czasy z dystr. empir.

Obliczenie czasu zakończenia:

$CzasZak := Czas + CzasTr$

9. Skasowanie w buforze zgłoszeń w toku, danych o zgłoszeniach, które się skończyły (ich czas zakończenia upłynął przed czasem nadejścia bieżącego zgłoszenia)

2

2

10. Utworzenie tablicy węzłów zakłócanych: WezZak[i] dla $i=1,\ldots,n_{sz}$

gdzie: n_{sz} - liczba stacji zakłócających.

$$WezZak[i] = \begin{cases} 0 & \text{gdy i-ta stacja nie pracuje} \\ k & \text{gdy i-ta stacja zakłóca k-ty węzeł} \end{cases}$$

11. Zapisanie do bufora zgłoszeń w toku, danych o bieżącym zgłoszeniu:
- czas nadejścia;
- czas trwania;
- rodzaj emisji;
- zakres;
- numer węzła łączności.

12. $i_{sz} := 1$
PNZ := 0
war := FALSE
CzRS := FALSE

13. SZak := <numer i_{sz}-tej stacji zakłóceń, co do odległości>

14. Czy stacja "SZak" może zakłócać zakres (KF, UKF, R/L) bieżącego zgłoszenia? — N

T

15. Czy stacja "SZak" ma wystarczający czas reakcji? — N

T

16. CzRS := TRUE

17. Czy stacja "SZak" ma moc wystarczającą do zakłócenia bieżącego zgłosz. — N

T

18. war := TRUE

19. Czy stacja "SZak" jest zajęta zakłócaniem węzła o wyższym priorytecie? — T

N

20. Zwiększenie o 1 liczników:
- ogólnej liczby zgłoszeń zakłóconych;
- zgłoszeń zakłóconych w danym zakresie.

Zapamiętanie:
WezZak[SZak] := NvWez
ZajSZ[SZak] := CzasZak

Dopisanie w buforze zgłoszeń w toku, numeru stacji zakłócającej dane zgłoszenie (SZak).

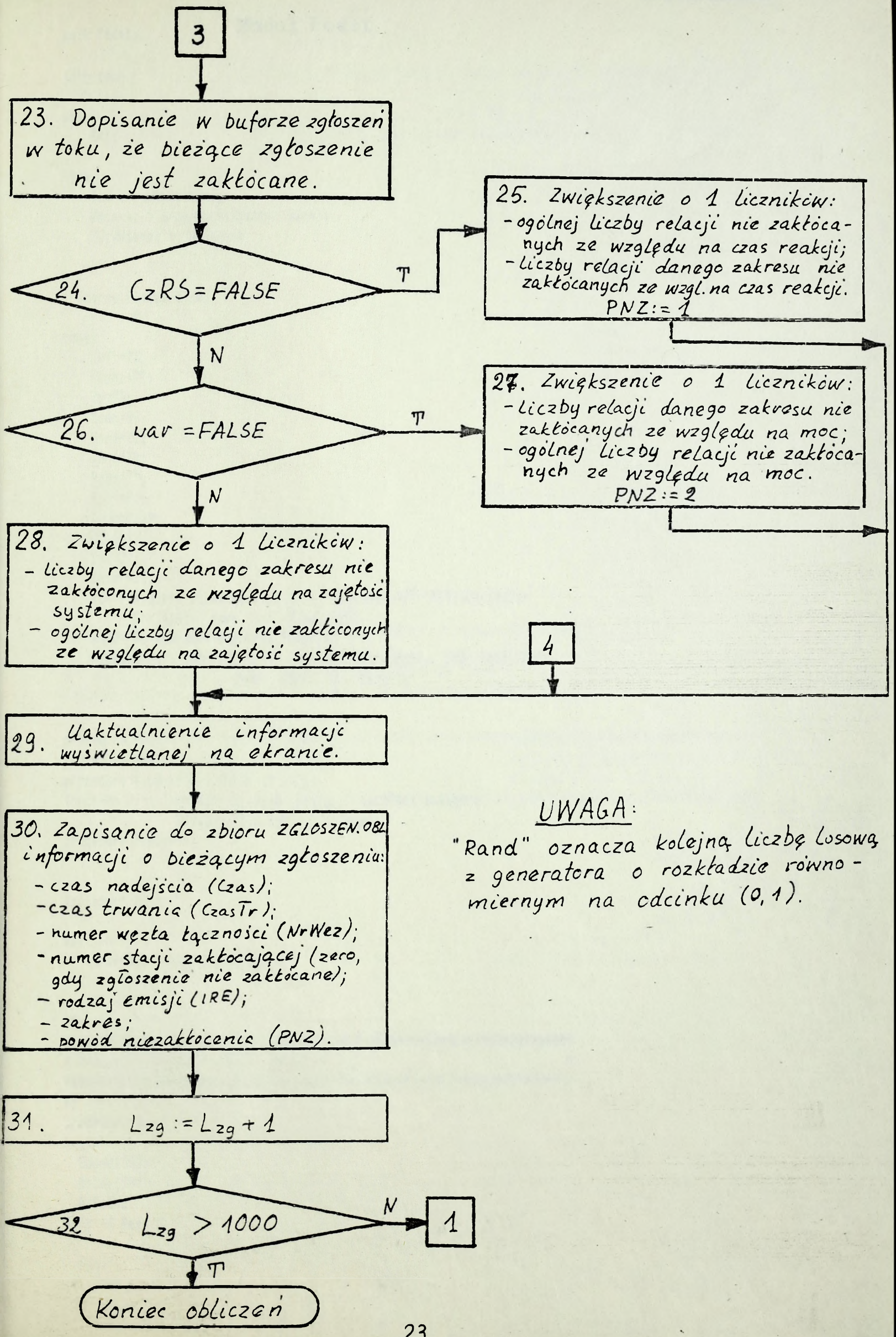
4

21. $i_{sz} := i_{sz} + 1$

22 $i_{sz} > n_{sz}$ — N

T

3



UWAGA:

"Rand" oznacza kolejną liczbę losową z generatora o rozkładzie równomiernym na odcinku (0, 1).

Moduł Podst

```
unit Podst;

interface

uses
    Crt;

type
    ZbInt= set of byte;
    WWiersz = procedure(NrWier: byte);
    RefWWiersz = ^WWiersz;

var
  Czarny,Bialy: byte;

const
    Left=75;
    Right=77;
    Up=72;
    Down=80;
    Home=71;
    _End=79;
    PgUp=73;
    PgDn=81;
    CHome=119;
    CEnd=117;
    Esc=27;
    Enter=13;
    kom1='Wybierz uzywajac klawiszy: '+Chr($1B)+Chr($19)+Chr($18)+
         Chr($1A)+' Home, End, PgUp, PgDn.';
    kom2='Po wybraniu nacisnij Enter.';
    KlawSterKur: ZbInt=[Left, Right, Up, Down, Home, _End, PgUp, PgDn,
                        CHome, CEnd, Esc, Enter];

procedure Beep;
procedure Maska;
procedure WypDyr(lan1,lan2: string);
function Wybor(WyswWiersz: RefWWiersz;  LiczPoz: integer;
               Odswiez: boolean):  integer;
procedure PiszeSie;


implementation

var
    StaryWyb, NowyWyb: integer;


(*
##############################################################################
# Procedura generuje sygnal dzwiekowy                                        #
##############################################################################
*)
procedure Beep;
begin
  Sound(500);
  Delay(300);
  NoSound
end;  { Beep }
```

```
(*
###############################################################################
# Procedura obramowuje ekran ramka - prostokatem                              #
# i na gorze ramki wyprowadza napis KSO ASG                                   #
###############################################################################
*)
procedure Maska;
  var
    i: integer;
  begin
     ClrScr;
     for i:=2 to 23 do
       begin
          GoToXY(1,i);
          Write(Chr($BA));
          GoToXY(80,i);
          Write(Chr($BA))
       end;
     GoToXY(1,1);
     Write(Chr($C9));
     for i:=2 to 79 do
       Write(Chr($CD));
     Write(Chr($BB));
     GoToXY(1,24);
     Write(Chr($C8));
     for i:=2 to 79 do
       Write(Chr($CD));
     Write(Chr($BC));
     GoToXY(15,1);
     Write(' KSO ASG WP ')
  end;


(*
###############################################################################
#  Procedura wyprowadza na ekran teksty                                       #
#  lan1 w linii 22 i lan2 w linii 23                                          #
###############################################################################
*)
procedure WypDyr;
  var i: integer;
  begin
    HighVideo;
    GoToXY(2,22);
    for i:=2 to 79 do Write(' ');
    GoToXY((80-Length(lan1)) div 2,22);
    Write(lan1);
    GoToXY(2,23);
    for i:=2 to 79 do Write(' ');
    GoToXY((80-Length(lan2)) div 2,23);
    Write(lan2);
    NormVideo
  end;


(*
###############################################################################
# Procedura czyta z klawiatury klawisz sterowania kursorem i pakuje           #
# do "j". Innych klawiszy nie przyjmuje.                                      #
###############################################################################
*)
procedure CzytKlawSK(var j: longint);
```

```
var Ch: char;
    b: byte;
begin
   j:=0;
   while j=0 do
     begin
       Ch:=ReadKey;
       j:=Ord(Ch);
       if j=0 then
         begin
           Ch:=ReadKey;
           j:=Ord(Ch)
         end
       else
         if j<>13 then j:=0;
       b:=j;
       if not (b in KlawSterKur) then
         begin
           j:=0;
           Sound(300);
           Delay(200);
           NoSound
         end
     end
end;



(*
#############################################################################
# Procedura wykonuje ruch kursorem /podswietleniem/ w menu, zgodnie         #
# z nacisnietym klawiszem sterowania kursorem /wartosc w "j"/.              #
#############################################################################
*)
procedure WykRuch(WyswWiersz: RefWWiersz;
                  LiczPoz: integer; j: longint);
  var b: byte;
  begin
     b:=j;
     case b of
          Up,Left: NowyWyb:=StaryWyb-1;
          Down,Right: NowyWyb:=StaryWyb+1;
          Home,PgUp,CHome: NowyWyb:=1;
          _End,PgDn,CEnd: NowyWyb:=LiczPoz
     end;
     if NowyWyb<1 then NowyWyb:=LiczPoz;
     if NowyWyb>LiczPoz then NowyWyb:=1;
     NormVideo;
     WyswWiersz^(StaryWyb);
     HighVideo;
     WyswWiersz^(NowyWyb);
     NormVideo;
     StaryWyb:=NowyWyb
  end;


(*
#############################################################################
# Funkcja sterujaca wyborem z menu /teksty oferty w Tekst, liczba           #
# pozycji w LiczPoz/. Wartoscia funkcji jest numer wybranej pozycji.        #
#############################################################################
*)
```

```
function Wybor;
  var
     i: byte;
     j: longint;
  begin
    if Odswiez then
      begin
        Maska;
        WypDyr(kom1,kom2);
      end;
    NormVideo;
    for i:=2 to LiczPoz do
      WyswWiersz^(i);
    NowyWyb:=1;
    HighVideo;
    WyswWiersz^(NowyWyb);
    StaryWyb:=NowyWyb;
    NormVideo;
    j:=0;
    while j<>13 do
      begin
        CzytKlawSK(j);
        WykRuch(WyswWiersz, LiczPoz, j)
      end;
    Wybor:=StaryWyb
  end;


{*
#########################################################################
# Procedura do "korkowania" wywolan nie napisanych procedur.            #
#########################################################################
*}
procedure PiszeSie;
  var Ch: char;
  begin
     WypDyr('Procedura w opracowaniu.','Nacisnij klawisz!');
     Ch:=ReadKey
  end;

end.
```

Moduł WRE1Decl

```
unit WRE1Decl;

interface


const
     MaxWL=100;
     MaxSZ=100;
     MaxZgl=300;
     NazwyZakr: array[1..3] of string[3] = ('UKF', 'KF ', 'R/1' );

type
  Zakr=(UKF, KF, RL);
  Wezel = record
            Przynal: string[10];
            Nazwa:   string[10];
            X,Y:     integer;
            Prior:       byte;
            LRK, LRU, LRR: byte;
            DLM, DMA: integer;
            A:           real;
            PK, PU, PR: integer;
            RP:          byte;
          end;
  SrodZakl = record
               Przynal: string[10];
               Nazwa:   string[10];
               X, Y:    integer;
               P:       integer;
               CzR:     byte;
               Zakres:  byte;
             end;
  Zgloszenie = record
                 CzNZ, CzTr: real;
                 NWL, NSZ, REM: byte;
                 ZZgl: Zakr;
                 PowodNZ: byte;
               end;
  NazwaStrony=(Czerwoni, Niebiescy);
  TypyObiektow=(WezlyLacz, SrodkiZakl);
  TypDzialania=(Dopisywanie, Poprawianie, Wyswietlanie, Drukowanie);
  FileWezly = file of Wezel;
  FileSrodZakl = file of SrodZakl;
  FileZgl = file of Zgloszenie;


var
  NrWar, ZbNazw, ZbWart: string;
  Strona: NazwaStrony;
  Obiekty: TypyObiektow;
  Dzialanie: TypDzialania;
  Kontynuowac: char;


   LiczbaWezlow, LiczbaSrodZakl: byte;
   TabWL: array[1..MaxWL] of Wezel;
   TabSZ: array[1..MaxSZ] of SrodZakl;


   LZg, LiczZglZakl, LiczNWar, LiczNZaj, LiczNCzR: word;
   TLZg, TLiczZglZakl, TLiczNWar, TLiczNZaj, TLiczNCzR:
                                  array[Zakr] of word;
```

```
  NazwaZb: string;
  NazwaZbStarego :string;
  LiczbaRekZb: integer;
  KoniecZb: boolean;
  Nadrzed: string[10];
  RekOK: char;
  BladWeWy: boolean;
  JestRek: boolean;

  ZbWezlowStary: FileWezly;
  ZbWezlow: FileWezly;
  RekWezel: Wezel;
  RekWezelStary: Wezel;

  ZbSrodZaklStary: FileSrodZakl;
  ZbSrodZakl: FileSrodZakl;
  RekSrodZakl: SrodZakl;
  RekSrodZaklStary: SrodZakl;

  ZbZgloszen: FileZgl;
  RekZgl: Zgloszenie;

implementation

end.
```

Moduł WRE2ZakB

```
unit WRE2ZakB;

interface

uses
   WRE1Decl, Podst, Crt;

procedure DrukWWL(i: byte);
procedure DrukWarWL(i: byte);
procedure CzytWWL(i: byte);
procedure CzytajWL(k: byte);
procedure DopWL;
procedure DrukWSZ(i: byte);
procedure DrukWarSZ(i: byte);
procedure CzytWSZ(i: byte);
procedure CzytajSZ(k: byte);
procedure DopSZ;


implementation

(*
##############################################################################
#  Procedura druk. nazwe wielkosci o podanym numerze/wprowadzanie Wez.Lacz#
##############################################################################
*)
procedure DrukWWL(i: byte);
begin
  with RekWezel do
  begin
    GotoXY(5,3+i);
    case i of
         1: Write('           Nazwa jednostki (maks. 10 znakow): ');
         2: Write('     Nazwa wezla lacznosci (maks. 10 znakow): ');
         3: Write('           Wspolrzedna X wezla lacznosci: X = ');
         4: Write('           Wspolrzedna Y wezla lacznosci: Y = ');
         5: Write('           Priorytet wezla lacznosci: Prior = ');
         6: Write('     Liczba relacji KF wezla lacznosci: LRK = ');
         7: Write('    Liczba relacji UKF wezla lacznosci: LRU = ');
         8: Write('Liczba relacji radiol. wezla lacznosci: LRR = ');
         9: Write('   Minimalna dlugosc linii lacznosci: DLM = ');
        10: Write('  Maksymalna dlugosc linii lacznosci: DMA = ');
        11: Write('  Srednie natezenie ruchu erlangach:   A = ');
        12: Write('                 Moc promieniowania KF: PK = ');
        13: Write('                Moc promieniowania UKF: PU = ');
        14: Write('          Moc promieniowania radiolinii: PR = ');
        15: Write('                 ZAPISAC DO ZBIORU (T/N) ? : ');
        16: Write('     CZY WPROWADZASZ NASTEPNY WEZEL (T/N) ? : ')
    end;  { case }
  end;  { with }
end;  { DrukWWL }



(*
##############################################################################
#  Procedura drukujaca wielkosc o podanym numerze /wprowadzanie Wez.Lacz. #
##############################################################################
*)
```

```
procedure DrukWarWL(i: byte);
begin
  GotoXY(52, 3+i);
  with RekWezel do
  case i of
       1: Write(Przynal :10);
       2: Write(Nazwa :10);
       3: Write(X :10);
       4: Write(Y :10);
       5: Write(Prior :10);
       6: Write(LRK :10);
       7: Write(LRU :10);
       8: Write(LRR :10);
       9: Write(DLM :10);
      10: Write(DMA :10);
      11: Write(A :10:3);
      12: Write(PK :10);
      13: Write(PU :10);
      14: Write(PR :10);
      15: Write(RekOK :10);
      16: Write(Kontynuowac :10)
  end;  { case, with }
end;  { DrukWarWL }



(*
##############################################################################
#  Procedura czytajaca wielkosc o podanym numerze /czytanie Wez. Lacz./      #
##############################################################################
*)
procedure CzytWWL(i: byte);
var DanOK: boolean;
begin
  Window(52, 3+i, 62, 3+i);
  DanOK:=FALSE;
  while not DanOK do
  begin
{$I-}
    with RekWezel do
    case i of
         1: Readln(Przynal);
         2: Readln(Nazwa);
         3: Readln(X);
         4: Readln(Y);
         5: Readln(Prior);
         6: Readln(LRK);
         7: Readln(LRU);
         8: Readln(LRR);
         9: Readln(DLM);
        10: Readln(DMA);
        11: Readln(A);
        12: Readln(PK);
        13: Readln(PU);
        14: Readln(PR);
        15: Readln(RekOK);
        16: Readln(Kontynuowac)
    end;  { case, with }
{$I+}
    if IOResult<>0 then
    begin
```

```
      ClrScr;
      Beep;
      DanOK:=FALSE;
    end
    else
      DanOK:=TRUE;
  end;  { while }

  Window(1, 1, 80, 25);
  DrukWarWL(i);
end;  { CzytWWL }


(*
##############################################################################
#  Procedura czytajaca dane o Wezle Lacznosci.                               #
##############################################################################
*)
procedure CzytajWL(k: byte);
var
   i: byte;
   Kolor: string[11];
begin
  Maska;
  case Strona of
      Czerwoni:  Kolor:='CZERWONYCH ';
      Niebiescy: Kolor:='NIEBIESKICH';
  end;  { case }
  WypDyr('Dopisywanie danych o wezlach lacznosci '+Kolor,
         'Kazda dana koncz nacisnieciem klawisza Enter.');
  NormVideo;
  for i:=1 to k do
  begin
    DrukWWL(i);
    CzytWWL(i)
  end  { for }
end;  { CzytajWL }


(*
##############################################################################
#  Procedura dopisujaca dane do zbioru danych o Wezlach Lacznosci.           #
##############################################################################
*)
procedure DopWL;
begin
  Assign(ZbWezlow, NazwaZb);
{$I-}
  Reset(ZbWezlow);
  if IOResult<>0 then
    Rewrite(ZbWezlow);
{$I+}
  Seek(ZbWezlow,FileSize(ZbWezlow));

  Kontynuowac:='T';
  repeat
    CzytajWL(16);
    if UpCase(RekOK)='T' then  Write(ZbWezlow, RekWezel);
  until UpCase(Kontynuowac)='N';
  Close(ZbWezlow)
```

```
end;  { DopWL }


(*
##############################################################################
#  Procedura druk. nazwe wielkosci o podanym numerze /wprow. Srod. Zakl./ #
##############################################################################
*)
procedure DrukWSZ(i: byte);
begin
  with RekSrodZakl do
  begin
    GotoXY(5,5+i);
    case i of
         1: Write('           Nazwa jednostki (maks. 10 znakow): ');
         2: Write('     Nazwa srodka zaklocen (maks. 10 znakow): ');
         3: Write('           Wspolrzedna X srodka zaklocen: X = ');
         4: Write('           Wspolrzedna Y srodka zaklocen: Y = ');
         5: Write('                    Moc srodka zaklocen: P = ');
         6: Write('   Czas reakcji srodka zaklocen (sek.): CzR = ');
         7: Write('                Zakres (1-UKF, 2-KF, 3-R/1) : ');
         8: Write('                  ZAPISAC DO ZBIORU (T/N) ? : ');
         9: Write('CZY WPROWADZASZ NASTEPNY SROD. ZAKL.(T/N) ? : ')
    end;   { case }
  end;  { with }
end;  { DrukWSZ }


(*
##############################################################################
#  Procedura drukujaca wielkosc o podanym numerze /wprowadzanie Srod.Zak. #
##############################################################################
*)

procedure DrukWarSZ(i: byte);
begin
  GotoXY(52, 5+i);
  with RekSrodZakl do
  case i of
       1: Write(Przynal :10);
       2: Write(Nazwa :10);
       3: Write(X :10);
       4: Write(Y :10);
       5: Write(P :10);
       6: Write(CzR :10);
       7: Write(NazwyZakr[Zakres] :10);
       8: Write(RekOK :10);
       9: Write(Kontynuowac :10)
  end;  { case, with }
end;  { DrukWarSZ }



(*
##############################################################################
#  Procedura czytajaca wielkosc o podanym numerze /wprow. Srod. Zakl./    #
##############################################################################
*)
procedure CzytWSZ(i: byte);
var DanOK: boolean;
```

```
begin
  Window(52, 5+i, 62, 5+i);
  DanOK:=FALSE;
  while not DanOK do
  begin
{$I-}
    with RekSrodZakl do
    case i of
         1: Readln(Przynal);
         2: Readln(Nazwa);
         3: Readln(X);
         4: Readln(Y);
         5: Readln(P);
         6: Readln(CzR);
         7: Readln(Zakres);
         8: Readln(RekOK);
         9: Readln(Kontynuowac)
    end;  { case, with }
{$I+}
    if (IOResult<>0) or ((i=7) and ((RekSrodZakl.Zakres>3) or
                                    (RekSrodZakl.Zakres<1))) then
    begin
      ClrScr;
      Beep;
      DanOK:=FALSE;
    end
    else
      DanOK:=TRUE;
  end;  { while }

  Window(1, 1, 80, 25);
  DrukWarSZ(i);
end;  { CzytWSZ }


(*
##############################################################################
#  Procedura czytajaca dane o Srodku Zaklocen.                               #
##############################################################################
*)
procedure CzytajSZ(k: byte);
var
   i: byte;
   Kolor: string[11];
begin
  Maska;
  case Strona of
       Czerwoni:  Kolor:='CZERWONYCH ';
       Niebiescy: Kolor:='NIEBIESKICH';
  end;  { case }
  WypDyr('Wprowadzanie danych o srodkach zaklocen '+Kolor,
         'Kazda dana koncz nacisnieciem klawisza Enter.');
  NormVideo;
  for i:=1 to k do
  begin
    DrukWSZ(i);
    CzytWSZ(i)
  end
end;  { CzytajSZ }
```

```
{*
#############################################################################
#  Procedura dopisujaca dane do zbioru o Srodkach Zaklocen.                 #
#############################################################################
*}
procedure DopSZ;
begin
  Assign(ZbSrodZakl, NazwaZb);
{$I-}
  Reset(ZbSrodZakl);
  if IOResult<>0 then  Rewrite(ZbSrodZakl);
{$I+}
  Seek(ZbSrodZakl,FileSize(ZbSrodZakl));

  Kontynuowac:='T';
  repeat
    CzytajSZ(9);
    if UpCase(RekOK)='T' then  Write(ZbSrodzakl, RekSrodZakl);
  until UpCase(Kontynuowac)='N';
  Close(ZbSrodZakl)
end;  { DopSZ }


end.  { unit WREZakB }
```

Moduł WRE3AktB

```
unit WRE3AktB;


interface

uses
  WRE2ZakB, WRE1Decl, Podst, Crt;

procedure WEdWezel(NrWier: byte);
procedure WPopWL(NrWier: byte);
procedure PopWL;
procedure EdycjaWezlow;
procedure WEdSrodZakl(NrWier: byte);
procedure WPopSZ(NrWier: byte);
procedure PopSZ;
procedure EdycjaSrodZakl;

implementation


(*
##########################################################################
#  Procedura przeglada wezly lacznosci biezacej jednostki i ew. je      #
#  przepisuje.                                                           #
##########################################################################
*)
procedure PrzejdzJednWL(Przepisywac: boolean);
begin
  KoniecZb:=FALSE;
  while (not KoniecZb) and (Nadrzed=RekWezelStary.Przynal) do
  begin
    if Przepisywac then  Write(ZbWezlow, RekWezelStary);
    if FilePos(ZbWezlowStary) < FileSize(ZbWezlowStary) then
      Read(ZbWezlowStary, RekWezelStary)
    else
      KoniecZb:=TRUE;
  end;  { while }
  if (not KoniecZb) then
    Nadrzed:=RekWezelStary.Przynal
  else
    Kontynuowac:='N';
end;  { ZachowajWezly }



(*
##########################################################################
#  Procedura przepisuje zbior wezlow lacznosci do konca.                 #
##########################################################################
*)
procedure PrzepDoKoncaWL;
begin
  KoniecZb:=FALSE;
  while (not KoniecZb) do
  begin
    Write(ZbWezlow, RekWezelStary);
    if FilePos(ZbWezlowStary) < FileSize(ZbWezlowStary) then
      Read(ZbWezlowStary, RekWezelStary)
    else
```

```
      KoniecZb:=TRUE
  end;  { while }
  Kontynuowac:='N'
end;  { PrzepDoKoncaWL }



(*
#########################################################################
#  Procedura wyswietla dane o biezacym wezle lacznosci.                 #
#########################################################################
*)
procedure WyswWezel;
var i: byte;
begin
  for i:=1 to 14 do
  begin
    DrukWWL(i);
    DrukWarWL(i)
  end
end;  { WyswWezel }


(*
#########################################################################
#  Procedura wprowadzajaca zmiany w wezle                               #
#########################################################################
*)
procedure ZmienWezel;
var
  Druk: WWiersz;
  i: byte;
begin
  Kontynuowac:='N';
  RekOK:='T';
  Maska;
  WypDyr(koml,
      'Nastepnie nacisnij Enter, wprowadz wartosc i nacisnij Enter');
  for i:=1 to 15 do
  begin
    DrukWWL(i);
    DrukWarWL(i)
  end;  { for }
  GotoXY(5,19);
  Write('CZY ZAKONCZYLES EDYCJE WEZLA (T/N) ?');
  DrukWarWL(16);
  Druk:=DrukWarWL;
  while UpCase(Kontynuowac)<>'T' do
  begin
    i:=Wybor(@@Druk, 16, FALSE);
    CzytWWL(i)
  end;  { while }
end;  { ZmienWezel }



(*
#########################################################################
#  Procedura drukujaca dla procedury EdycjaWezlow /Wybor/.              #
#########################################################################
```

```
*)
procedure WEdWezel(NrWier: byte);
begin
  GotoXY(15*NrWier, 20);
  case NrWier of
      1: Write('ZACHOWAC');
      2: Write('SKASOWAC');
      3: Write('ZMIENIAC');
      4: Write('DOPISAC PRZED')
  end;  { case }
end;  { WEdWezel }



(*
#########################################################################
#  Procedura sterujaca edycja wezlow lacznosci biezacej jednostki.  #
#########################################################################
*)
procedure EdycjaWezlow;
var
  wariant: byte;
  Druk: WWiersz;
begin
  KoniecZb:=FALSE;
  while (not KoniecZb) and (Nadrzed=RekWezelStary.Przynal) do
  begin
    Maska;
    RekWezel:=RekWezelStary;
    WyswWezel;
    WypDyr(kom1, kom2);
    Druk:=WEdWezel;
    wariant:=Wybor(@@Druk, 4, FALSE);
    JestRek:=FALSE;
    case wariant of
        1: Write(ZbWezlow, RekWezel);
        2: ;
        3: begin
             ZmienWezel;
             if UpCase(RekOK)='T' then
                Write(ZbWezlow, RekWezel)
           end;

        4: begin
             CzytajWL(15);
             if UpCase(RekOK)='T' then
                Write(ZbWezlow, RekWezel);
             JestRek:=TRUE;
           end;
    end;  { case }
    if (not JestRek) then
      if FilePos(ZbWezlowStary) < FileSize(ZbWezlowStary) then
        Read(ZbWezlowStary, RekWezelStary)
      else
        KoniecZb:=TRUE;
  end;  { while }
  if (not KoniecZb) then
    Nadrzed:=RekWezelStary.Przynal
  else
    Kontynuowac:='N';
end;  { EdycjaWezlow }
```

```
{*
#############################################################################
#  Procedura przydzielajaca zbiory Bazowe Wezlow Lacznosci                  #
#  Ustawia "BladWeWy"                                                       #
#############################################################################
*}
procedure PrzydzBWL;
var
  ch: char;
begin
{$I-}
  Assign(ZbWezlowStary, NazwaZbStarego);
  Erase(ZbWezlowStary);
  Assign(ZbWezlowStary, NazwaZb);
  Rename(ZbWezlowStary, NazwaZbStarego);
  if IOResult<>0 then ;
  Assign(ZbWezlowStary, NazwaZbStarego);
  Reset(ZbWezlowStary);
  BladWeWy:=TRUE;
  if (IOResult=0) then
    if (FileSize(ZbWezlowStary)>0) then
      BladWeWy:=FALSE
    else
      Close(ZbWezlowStary);
{$I+}
  Assign(ZbWezlow, NazwaZb);
  Rewrite(ZbWezlow);
  if BladWeWy then
  begin
    WypDyr('Brak zbioru do poprawiania', 'Nacisnij klawisz');
    ch:=ReadKey;
    if Ord(ch)=0 then ch:=ReadKey
  end;  { if }
end;  { PrzydzBWL }



{*
#############################################################################
#  Procedura drukujaca dla procedury PopWL /Wybor/.                         #
#############################################################################
*}
procedure WPopWL(NrWier: byte);
begin
  GotoXY(5, 10+NrWier);
  case NrWier of
      1: Write('Zachowanie wezlow lacznosci danej jednostki');
      2: Write('Skasowanie wezlow lacznosci danej jednostki');
      3: Write('Poprawianie wezlow lacznosci danej jednostki');
      4: Write('Zachowanie wezlow lacznosci reszty jednostek');
      5: Write('Skasowanie wezlow lacznosci reszty jednostek');
  end;  { case }
end;  { WPopWL }



{*
#############################################################################
#  Procedura glowna, sterujaca poprawianiem zbioru wezlow lacznosci.        #
```

```
{#############################################################################
}
procedure PopWL;
var
  wariant: byte;
  Druk: WWiersz;
begin
  PrzydzBWL;
  if (not BladWeWy) then
  begin
    Read(ZbWezlowStary, RekWezelStary);
    Nadrzed:=RekWezelStary.Przynal;
    Kontynuowac:='T';
    while UpCase(Kontynuowac)<>'N' do
    begin
      Maska;
      GotoXY(5, 5);
      Write('Wezly lacznosci: '+Nadrzed);
      WypDyr(kom1, kom2);
      Druk:=WPopWL;
      wariant:=Wybor(@@Druk, 5, FALSE);
      case wariant of
           1: PrzejdzJednWL(TRUE);
           2: PrzejdzJednWL(FALSE);
           3: EdycjaWezlow;
           4: PrzepDoKoncaWL;
           5: Kontynuowac:='N';
      end;  { case }
    end;  { while }
    Close(ZbWezlow);
    Close(ZbWezlowStary)
  end  { if }
end;  { PopWL }



{*
#############################################################################
#  Procedura przeglada srodki zaklocen biezacej jednostki i ew. je  #
#  przepisuje.                                                              #
#############################################################################
}
procedure PrzejdzJednSZ(Przepisywac: boolean);
begin
  KoniecZb:=FALSE;
  while (not KoniecZb) and (Nadrzed=RekSrodZaklStary.Przynal) do
  begin
    if Przepisywac then  Write(ZbSrodZakl, RekSrodZaklStary);
    if FilePos(ZbSrodZaklStary) < FileSize(ZbSrodZaklStary) then
      Read(ZbSrodZaklStary, RekSrodZaklStary)
    else
      KoniecZb:=TRUE;
  end;  { while }
  if (not KoniecZb) then
    Nadrzed:=RekSrodZaklStary.Przynal
  else
    Kontynuowac:='N';
end;  { PrzejdzSrodZakl }
```

```
(*
#############################################################################
#  Procedura przepisuje zbior srodkow zaklocen do konca.                    #
#############################################################################
*)
procedure PrzepDoKoncaSZ;
begin
  KoniecZb:=FALSE;
  while (not KoniecZb) do
  begin
    Write(ZbSrodZakl, RekSrodZaklStary);
    if FilePos(ZbSrodZaklStary) < FileSize(ZbSrodZaklStary) then
      Read(ZbSrodZaklStary, RekSrodZaklStary)
    else
      KoniecZb:=TRUE
  end;  { while }
  Kontynuowac:='N'
end;  { PrzepDoKoncaSZ }



(*
#############################################################################
#  Procedura wyswietla dane o biezacym srodku zaklocen.                     #
#############################################################################
*)
procedure WyswSrodZakl;
var i: byte;
begin
  for i:=1 to 7 do
  begin
    DrukWSZ(i);
    DrukWarSZ(i)
  end
end;  { WyswSrodZakl }


(*
#############################################################################
#  Procedura wprowadzajaca zmiany w Srodku Zaklocen                         #
#############################################################################
*)
procedure ZmienSrodZakl;
var
  Druk: WWiersz;
  i: byte;
begin
  Kontynuowac:='N';
  RekOK:='T';
  Maska;
  WypDyr(kom1,
      'Nastepnie nacisnij Enter, wprowadz wartosc i nacisnij Enter');
  for i:=1 to 8 do
  begin
    DrukWSZ(i);
    DrukWarSZ(i)
  end;  { for }
  GotoXY(5,14);
  Write('CZY ZAKONCZYLES EDYCJE SRODKA ZAKL. (T/N) ? :');
  DrukWarSZ(9);
  Druk:=DrukWarSZ;
  while UpCase(Kontynuowac)<>'T' do
```

```
  begin
    i:=Wybor(@@Druk, 9, FALSE);
    CzytWSZ(i)
  end;  { while }
end;  { ZmienSrodZakl }



(*
##############################################################################
#  Procedura drukujaca dla procedury EdycjaSrodZakl /Wybor/.                 #
##############################################################################
*)
procedure WEdSrodZakl(NrWier: byte);
begin
  GotoXY(15*NrWier, 20);
  case NrWier of
      1: Write('ZACHOWAC');
      2: Write('SKASOWAC');
      3: Write('ZMIENIAC');
      4: Write('DOPISAC PRZED')
  end;  { case }
end;  { WEdSrodZakl }



(*
##############################################################################
#  Procedura sterujaca edycja srodkow zaklocen biezacej jednostki.           #
##############################################################################
*)
procedure EdycjaSrodZakl;
var
  wariant: byte;
  Druk: WWiersz;
begin
  KoniecZb:=FALSE;
  while (not KoniecZb) and (Nadrzed=RekSrodZaklStary.Przynal) do
  begin
    Maska;
    RekSrodZakl:=RekSrodZaklStary;
    WyswSrodZakl;
    WypDyr(kom1, kom2);
    Druk:=WEdSrodZakl;
    wariant:=Wybor(@@Druk, 4, FALSE);
    JestRek:=FALSE;
    case wariant of
        1: Write(ZbSrodZakl, RekSrodZakl);
        2: ;
        3: begin
             ZmienSrodZakl;
             if UpCase(RekOK)='T' then
                Write (ZbSrodZakl, RekSrodZAkl)
           end;

        4: begin
             CzytajSZ(8);
             if UpCase(RekOK)='T' then
                Write(ZbSrodZakl, RekSrodZakl);
             JestRek:=TRUE;
           end;
    end;  { case }
```

```
    if (not JestRek) then
      if FilePos(ZbSrodZaklStary) < FileSize(ZbSrodZaklStary) then
        Read(ZbSrodZaklStary, RekSrodZaklStary)
      else
        KoniecZb:=TRUE;
  end;  { while }
  if (not KoniecZb) then
    Nadrzed:=RekSrodZaklStary.Przynal
  else
    Kontynuowac:='N';
end;  { EdycjaSrodZakl }



{*
##########################################################################
#  Procedura przydzielajaca zbiory Bazowe Srodkow Zaklocen               #
#  Ustawia "BladWeWy"                                                    #
##########################################################################
*}
procedure PrzydzBSZ;
var ch: char;
begin
{$I-}
  Assign(ZbSrodZaklStary, NazwaZbStarego);
  Erase(ZbSrodZaklStary);
  Assign(ZbSrodZaklStary, NazwaZb);
  BladWeWy:=FALSE;
  Rename(ZbSrodZaklStary, NazwaZbStarego);
  if IOResult<>0 then ;
  Assign(ZbSrodZaklStary, NazwaZbStarego);
  Reset(ZbSrodZaklStary);
  BladWeWy:=TRUE;
  if (IOResult=0) then
    if (FileSize(ZbSrodZaklStary)>0) then
      BladWeWy:=FALSE
    else
      Close(ZbSrodZaklStary);
{$I+}
  Assign(ZbSrodZakl, NazwaZb);
  Rewrite(ZbSrodZakl);
  if BladWeWy then
  begin
    WypDyr('Brak zbioru do poprawiania', 'Nacisnij klawisz');
    ch:=ReadKey;
    if Ord(ch)=0 then ch:=ReadKey
  end;  { else }
end;   { PrzydzBSZ }



{*
##########################################################################
#  Procedura drukujaca dla procedury PopSZ /Wybor/.                      #
##########################################################################
*}
procedure WPopSZ(NrWier: byte);
begin
  GotoXY(5, 10+NrWier);
  case NrWier of
      1: Write('Zachowanie srodkow zaklocen danej jednostki');
```

```
      2: Write('Skasowanie srodkow zaklocen danej jednostki');
      3: Write('Poprawianie srodkow zaklocen danej jednostki');
      4: Write('Zachowanie srodkow zaklocen reszty jednostek');
      5: Write('Skasowanie srodkow zaklocen reszty jednostek');
  end;  { case }
end;  { WPopSZ }



{*
###########################################################################
#  Procedura glowna, sterujaca poprawianiem zbioru srodkow zaklocen.#
###########################################################################
*}
procedure PopSZ;
var
  wariant: byte;
  Druk: WWiersz;
begin
  PrzydzBSZ;
  if (not BladWeWy) then
  begin
    Read(ZbSrodZaklStary, RekSrodZaklStary);
    Nadrzed:=RekSrodZaklStary.Przynal;
    Kontynuowac:='T';
    while UpCase(Kontynuowac)<>'N' do
    begin
      Maska;
      GotoXY(5, 5);
      Write('Srodki zaklocen jednostki: '+Nadrzed);
      WypDyr(kom1, kom2);
      Druk:=WPopSZ;
      wariant:=Wybor(@@Druk, 5, FALSE);
      case wariant of
           1: PrzejdzJednSZ(TRUE);
           2: PrzejdzJednSZ(FALSE);
           3: EdycjaSrodZakl;
           4: PrzepDoKoncaSZ;
           5: Kontynuowac:='N';
      end;  { case }
    end;  { while }
    Close(ZbSrodZakl);
    Close(ZbSrodZaklStary)
  end  { if }
end;  { PopSZ }


end.   { unit WRE3AktB }
```

Moduł WRE4WypB

```
unit WRE4WypB;

interface

uses WREiDecl, Podst, Crt;

procedure WyswWL;
procedure DrukWL;
procedure WyswSZ;
procedure DrukSZ;


implementation

var
   ch: char;
   Wydruk: text;
   i: byte;
   PageSize: byte;
   Ekran: boolean;
   Konczyc: boolean;



(*
###############################################################
#  Procedura wyprowadzania wiersza Wez. Lacz.                 #
###############################################################
*)
procedure WyprWWL;
begin
  with RekWezel do
  Writeln(Wydruk, Przynal:10, Nazwa:11, X:5, Y:5,
                  Prior:3, LRK:3, LRU:3, LRR:3,
                  DLM:4, DMA:4, A:8:3,
                  PK:6, PU:6, PR:6);
end;  { WyprWWL }



(*
###############################################################
# Procedura wyprowadzania wiersza nagl. Wez. Lacz.            #
###############################################################
*)
procedure WyprNWL;
begin
  Writeln(Wydruk, ' Jednostka      Wezel     X     Y Pr ',
            'LK LU LR DLM DMA        A    PK    PU    PR');
end;  { WyprNWL }



(*
###############################################################
# Procedura wyprowadzania wiersza Srod. Zakl.                 #
###############################################################
*)
procedure WyprWSZ;
begin
  with RekSrodZakl do
  Writeln(Wydruk,
```

```
            Przynal:10, Nazwa:11, X:5, Y:5, P:6,
            CzR:6, NazwyZakr[Zakres]:6);
end;  { WyprWSZ }



(*
###############################################################
# Procedura wyprowadzania wiersza nagl. Srod. Zakl.           #
###############################################################
*)
procedure WyprNSZ;
begin
  Writeln(Wydruk, ' Jednostka  Sr. zakl.    X    Y   ',
                                      'Moc   Cz.r. Zakr.');
end;  {WyprNSZ }



(*
###############################################################
# Procedura przygotowujaca strone                             #
###############################################################
*)
procedure PrzygotStr;
begin
  if Ekran then
  begin
    Maska;
    WypDyr(' ','Nacisnij klawisz, Esc - koniec wyswietlania.');
    GoToXY(2,2);
  end
  else
  begin
    Write(Wydruk, Chr(12));
    Writeln;
    Writeln;
  end;  { if }
  case Obiekty of
      WezlyLacz:  WyprNWL;
      SrodkiZakl: WyprNSZ;
  end;  { case }
end;  { PrzygotStr }



(*
###############################################################
# Procedura oczekiwania po wyprow. strony na ekran            #
###############################################################
*)
procedure Czekaj;
begin
  if Ekran then
  begin
    ch:=ReadKey;
    if Ord(ch)=27 then
      Konczyc:=TRUE
    else
      if Ord(ch)=0 then
        ch:=ReadKey;
  end;  { if }
```

```
end;  { Czekaj }


(*
###############################################################
# Procedura wyprowadzania Wezlow Lacz.                        #
###############################################################
*)
procedure WyprowWL;
begin
  Konczyc:=FALSE;
  while (FilePos(ZbWezlow)<FileSize(ZbWezlow))
        and (not Konczyc) do
  begin
    PrzygotStr;
    for i:=1 to PageSize do
      if FilePos(ZbWezlow)<FileSize(ZbWezlow) then
      begin
        Read(ZbWezlow, RekWezel);
        if Ekran then GotoXY(2, i+3);
        WyprWWL;
      end;  { for, if }
    Czekaj;
  end;  { while }
end;  { WyprowWWL }


(*
###############################################################
# Procedura przydzielania zbioru Wez. Lacz.                   #
###############################################################
*)
procedure PrzydzWL;
begin
  Assign(ZbWezlow, NazwaZb);
{$I-}
  Reset(ZbWezlow);
  Konczyc:=TRUE;
  if (IOResult=0) then
    if (FileSize(ZbWezlow)>0) then
      Konczyc:=FALSE
    else
      Close(ZbWezlow);
{$I+}
end;  { PrzydzWL }


(*
###############################################################
# Procedura glowna wyswietlania Wez. Lacz.                    #
###############################################################
*)
procedure WyswWL;
begin
  PrzydzWL;
  if (not Konczyc) then
  begin
    AssignCrt(Wydruk);
    Rewrite(Wydruk);
```

```
    PageSize:=17;
    Ekran:=TRUE;
    WyprowWL;
    Close(Wydruk);
    Close(ZbWezlow)
  end
  else
  begin
    WypDyr('Brak zbioru: '+NazwaZb, 'Nacisnij klawisz');
    Ekran:=TRUE;
    Czekaj
  end;

end;  { WyswWL }



(*
###############################################################
# Procedura glowna drukowania Wez. Lacz.                      #
###############################################################
*)
procedure DrukWL;
begin
  PrzydzWL;
  if (not Konczyc) then
  begin
    Assign(Wydruk,'PRN');
    Rewrite(Wydruk);
    PageSize:=56;
    Ekran:=FALSE;
    WyprowWL;
    Close(Wydruk);
    Close(ZbWezlow);
  end
  else
  begin
    WypDyr('Brak zbioru: '+NazwaZb, 'Nacisnij klawisz');
    Ekran:=TRUE;
    Czekaj
  end;
end;  { DrukWL }



(*
###############################################################
# Procedura sterujaca wyprowadz. Srod. Zakl.                  #
###############################################################
*)
procedure WyprowSZ;
begin
  Konczyc:=FALSE;
  while (FilePos(ZbSrodZakl)<FileSize(ZbSrodZakl))
        and (not Konczyc) do
  begin
    PrzygotStr;
    for i:=1 to PageSize do
      if FilePos(ZbSrodZakl)<FileSize(ZbSrodZakl) then
      begin
        Read(ZbSrodZakl, RekSrodZakl);
        if Ekran then GotoXY(2, i+3);
```

```
      WyprWSZ;
    end;  { for, if }
   Czekaj;
  end;  { while }
end;  { WyprWSZ }


(*
#################################################################
# Procedura przydzielania zbioru Srod. Zakl.                    #
#################################################################
*)
procedure PrzydzSZ;
begin
  Assign(ZbSrodZakl, NazwaZb);
{$I-}
  Reset(ZbSrodZakl);
  Konczyc:=TRUE;
  if (IOResult=0) then
    if (FileSize(ZbSrodZakl)>0) then
      Konczyc:=FALSE
    else
      Close(ZbSrodZakl);
{$I+}
end;  { PrzydzSZ }


(*
#################################################################
# Procedura glowna wyswietlania Srodkow Zakl.                   #
#################################################################
*)
procedure WyswSZ;
begin
  PrzydzSZ;
  if (not Konczyc) then
  begin
    AssignCrt(Wydruk);
    Rewrite(Wydruk);
    PageSize:=17;
    Ekran:=TRUE;
    WyprowSZ;
    Close(Wydruk);
    Close(ZbSrodZakl)
  end
  else
  begin
    WypDyr('Brak zbioru: '+NazwaZb, 'Nacisnij klawisz');
    Ekran:=TRUE;
    Czekaj
  end;
end;  { WyswSZ }


(*
#################################################################
# Procedura glowna drukowania Srodkow Zakl.                     #
#################################################################
*)
```

```
procedure DrukSZ;
begin
  PrzydzSZ;
  if (not Konczyc) then
  begin
    Assign(Wydruk,'PRN');
    Rewrite(Wydruk);
    PageSize:=65;
    Ekran:=FALSE;
    WyprowSZ;
    Close(Wydruk);
    Close(ZbSrodZakl)
  end
  else
  begin
    WypDyr('Brak zbioru: '+NazwaZb, 'Nacisnij klawisz');
    Ekran:=TRUE;
    Czekaj
  end;
end;  { DrukSZ }

end.  { unit WRE4WypB }
```

Moduł WRE5Czyd

```
unit WRE5CzyD;

interface

uses
    WRE3AktB, WRE2ZakB, WRE1Decl, Podst, Crt;

procedure WprowWezLacz;
procedure WprowSrodZakl;

implementation


type
  Jedn = record
              Przynal: string[10];
              NrRek: word
         end;

var
  BrakBazy: boolean;
  FoczRek: byte;
  Licz, i, j: byte;
  WykJedn: array[1..100] of Jedn;
  RekJedn: Jedn;
  ch: char;
  Xfl, Yfl, Xfp, Yfp: integer;
  Xtl, Ytl, Xtp, Ytp: integer;
  Xf, Yf, Xt, Yt: integer;
  NazwaJedn: string[10];


{*
###############################################################
# Procedura czytajaca liczbe                                  #
###############################################################
*)
procedure CzytLiczbe(var l: integer;   x,y: byte);
var
  LiczOK: boolean;
begin
{$I-}
  LiczOK:=FALSE;
  repeat
    Window(x,y,x+4,y);
    ClrScr;
    Readln(l);
    if (IOResult<>0) or (Abs(l)>999) then
      Beep
    else
      LiczOK:=TRUE;
  until LiczOK=TRUE;
  Window(1,1,80,25);
  GotoXY(x,y);
  Write(l:4);
{$I+}
end;
```

```
(*
#################################################################
# Procedura czytajaca nazwe i polozenie jaednostki              #
#################################################################
*)
procedure CzytajNazwelPoloz;
begin
  Maska;
  WypDyr('Wprowadz podane wielkosci','Kazda koncz Enter');

  GotoXY(5,5);
  Write('Nazwa jednostki (brak - to bez zmian):');
  Window(22,5,31,5);
  Readln(NazwaJedn);
  Window(1,1,80,25);
  GotoXY(22,5);
  Write(NazwaJedn);

  GotoXY(5,7);
  Write('Wspolrzedne lewego skrzydla frontu jednostki:');
  GotoXY(5,8);
  Write('Xfl=           Yfl=');
  CzytLiczbe(Xfl,9,8);
  CzytLiczbe(Yfl,25,8);

  GotoXY(5,10);
  Write('Wspolrzedne prawego skrzydla frontu jednostki:');
  GotoXY(5,11);
  Write('Xfp=           Yfp=');
  CzytLiczbe(Xfp,9,11);
  CzytLiczbe(Yfp,25,11);

  GotoXY(5,13);
  Write('Wspolrzedne lewego skrzydla tylu jednostki:');
  GotoXY(5,14);
  Write('Xtl=           Ytl=');
  CzytLiczbe(Xtl,9,14);
  CzytLiczbe(Ytl,25,14);

  GotoXY(5,16);
  Write('Wspolrzedne prawego skrzydla tylu jednostki:');
  GotoXY(5,17);
  Write('Xtp=           Ytp=');
  CzytLiczbe(Xtp,9,17);
  CzytLiczbe(Ytp,25,17);

end;  { CzytajNazwelPoloz }



(*
#################################################################
# Procedura przeliczjaca wspolrzedne                            #
#################################################################
*)
procedure Wspolrzedne(var x, y: integer);
begin
  Xf:=Round(Xfl+((x/100.0)*(Xfp-Xfl)));
  Yf:=Round(Yfl+((x/100.0)*(Yfp-Yfl)));
  Xt:=Round(Xtl+((x/100.0)*(Xtp-Xtl)));
```

```
  Yt:=Round(Yt1+((x/100.0)*(Ytp-Yt1)));
  x:=Round(Xt+((y/100.0)*(Xf-Xt)));
  y:=Round(Yt+((y/100.0)*(Yf-Yt)));
end;  { Wspolrzedne }



(*
#################################################################
# Procedura drukujaca dla proc. wybierania WL z bazy            #
#################################################################
*)
procedure DrWybJednWL(i: byte);
begin
  GotoXY(5, 3+i);
  if (i>1) and (i<16) then
    if (PoczRek+i-2)>Licz then
      RekJedn.Przynal:='
    else
      RekJedn:=WykJedn[PoczRek+i-2];
  case i of
       1: Write(' ... poprzednie jednostki');
    2..15: Writeln(RekJedn.Przynal);
      16: Writeln('nastepne jednostki ...');
      17: Writeln('KONIEC WYBIERANIA Z BAZY')
  end;  { case }
end;  { DrWybJednWL }



(*
#################################################################
# Procedura wybierania Wez. Lacz. z bazy                        #
#################################################################
*)
procedure WybJednWL(var j: byte);
var
  Druk: WWiersz;
begin
  Druk:=DrWybJednWL;
  i:=1;
  while (i=1) or (i=16) or ((i<16) and ((PoczRek+i-2)>Licz)) do
  begin
    i:=Wybor(@@Druk, 17, TRUE);
    case i of
         1: if (PoczRek-14)>0 then PoczRek:=PoczRek-14
            else PoczRek:=1;
     2..15: j:=PoczRek+i-2;
        16: if (PoczRek+14)>Licz then PoczRek:=Licz
            else PoczRek:=PoczRek+14;
        17: Kontynuowac:='N'
    end;  { case }
  end;  { while }
end;  { WybJednWL }



(*
#################################################################
# Procedura przepisywania WL jednostki z bazy                   #
#################################################################
*)
```

```
procedure ZapiszJednWL;
var
  Pusty: boolean;
  i: byte;

begin
  CzytajNazwelPoloz;
  while ((RekWezelStary.Przynal)=Nadrzed) do
  begin
    Wspolrzedne(RekWezelStary.X, RekWezelStary.Y);

    Pusty:=TRUE;
    i:=1;
    while i<=Length(NazwaJedn) do
    begin
      if NazwaJedn[i]<>' ' then Pusty:=FALSE;
      i:=i+1;
    end;
    if not Pusty then RekWezelStary.Przynal:=NazwaJedn;

    Write(ZbWezlow, RekWezelStary);
    if (FilePos(ZbWezlowStary)<FileSize(ZbWezlowStary)) then
       Read(ZbWezlowStary, RekWezelStary)
    else
       Nadrzed:='@@@@@';
  end;
end;



(*
###################################################################
# Procedura sterujaca wybieraniem WL z bazy                       #
###################################################################
*)
procedure CzytajWLzBazy;
var
  i: word;

begin
  Licz:=0;
  Nadrzed:='@@@@@';
  while (Licz<100) and
        (FilePos(ZbWezlowStary)<FileSize(ZbWezlowStary)) do
  begin
    i:=FilePos(ZbWezlowStary);
    Read(ZbWezlowStary, RekWezel);
    if Nadrzed<>(RekWezel.Przynal) then
    begin
      Licz:=Licz+1;
      RekJedn.Przynal:=RekWezel.Przynal;
      RekJedn.NrRek:=i;
      WykJedn[Licz]:=RekJedn;
      Nadrzed:=RekWezel.Przynal;
    end;  { if }
  end;  { while }
  PoczRek:=1;
  Kontynuowac:='T';
  while UpCase(Kontynuowac)<>'N' do
  begin
    WybJednWL(j);
```

```
    if UpCase(Kontynuowac)<>'N' then
    begin
      RekJedn:=WykJedn[j];
      i:=RekJedn.NrRek;
      Seek(ZbWezlowStary, i);
      Read(ZbWezlowStary, RekWezelStary);
      Nadrzed:=RekWezelStary.Przynal;

      GotoXY(25,21);
      HighVideo;
      Write('CZY CHCESZ POPRAWIAC (T/N) ?');
      NormVideo;
      ch:=ReadKey;
      if UpCase(ch)='T' then
      begin
        EdycjaWezlow;
        Kontynuowac:='T'
      end
      else
      begin
        if Ord(ch)=0 then ch:=ReadKey;
        ZapiszJednWL
      end  { else }
    end;  { if }
  end;  { while }
end;  { OczytajWLzBazy }


(*
#################################################################
# Procedura  przydzielaj. i otwieraj. zbiory                    #
#################################################################
*)
procedure PrzydzZbWL;
begin
  Assign(ZbWezlow, NazwaZb);
{$I-}
  Reset(ZbWezlow);
  if IOResult<>0 then Rewrite(ZBWezlow);
{$I+}
  Seek(ZBWezlow, FileSize(ZbWezlow));
  Assign(ZbWezlowStary, NazwaZbStarego);
{$I-}
  Reset(ZbWezlowStary);
  BrakBazy:=TRUE;
  if (IOResult=0) then
    if (FileSize(ZbWezlowStary)>0) then
      BrakBazy:=FALSE
    else
      Close(ZbWezlowStary);
{$I+}

end;  { PrzydzZbWL }



(*
#################################################################
# Procedura sterujaca wprowadzaniem WL z bazy i consoli         #
#################################################################
*)
```

```
procedure WprowWezLacz;
begin
  PrzydzZbWL;
  if (not BrakBazy) then CzytajWLzBazy;
  Maska;
  GotoXY(5,22);
  Write('CZY CHCESZ DOPISYWAC (T/N) ?');
  Kontynuowac:=ReadKey;
  if Ord(Kontynuowac)=0 then ch:=ReadKey;
  while UpCase(Kontynuowac)='T' do
  begin
    CzytajWL(16);
    if UpCase(RekOK)='T' then Write(ZbWezlow, RekWezel);
  end;
  Close(ZbWezlow);
  if (not BrakBazy) then Close(ZbWezlowStary);
end;  { WprowWezLacz }



(*
###############################################################
# Procedura drukujaca dla proc. wybierania SZ z bazy          #
###############################################################
*)
procedure DrWybJednSZ(i: byte);
begin
  GotoXY(5, 3+i);
  if (i>1) and (i<16) then
    if (PoczRek+i-2)>Licz then
      RekJedn.Przynal:='
    else
      RekJedn:=WykJedn[PoczRek+i-2];
  case i of
       1: Write(' ... poprzednie jednostki');
    2..15: Writeln(RekJedn.Przynal);
      16: Writeln('nastepne jednostki ...');
      17: Writeln('KONIEC WYBIERANIA Z BAZY')
  end;  { case }
end;  { DrWybJednSZ }



(*
###############################################################
# Procedura wybierania Wez. Lacz. z bazy                      #
###############################################################
*)
procedure WybJednSZ(var i: byte);
var
  Druk: WWiersz;
begin
  Druk:=DrWybJednSZ;
  i:=1;
  while (i=1) or (i=16) or ((i<16) and ((PoczRek+i-2)>Licz)) do
  begin
    i:=Wybor(@@Druk, 17, TRUE);
    case i of
         1: if (PoczRek-14)>0 then PoczRek:=PoczRek-14
            else PoczRek:=1;
```

```
     2..15: j:=PoczRek+i-2;
        16: if (PoczRek+14)>Licz then PoczRek:=Licz
            else PoczRek:=PoczRek+14;
        17: Kontynuowac:='N'
    end;  { case }
  end;  { while }
end;  { WybJednSZ }


(*
#################################################################
# Procedura przepisywania SZ jednostki z bazy                   #
#################################################################
*)
procedure ZapiszJednSZ;
var
  Pusty: boolean;
  i: byte;
begin
  CzytajNazwePoloz;
  while (RekSrodZaklStary.Przynal=Nadrzed) do
  begin
    Wspolrzedne(RekSrodZaklStary.X, RekSrodZaklStary.Y);

    i:=1;
    Pusty:=TRUE;
    while i<=Length(NazwaJedn) do
    begin
      if NazwaJedn[i]<>' ' then Pusty:=FALSE;
      i:=i+1;
    end;
    if not Pusty then RekSrodZaklStary.Przynal:=NazwaJedn;

    Write(ZbSrodZakl, RekSrodZaklStary);
    if (FilePos(ZbSrodZaklStary)<FileSize(ZbSrodZaklStary)) then
       Read(ZbSrodZaklStary, RekSrodZaklStary)
    else
       Nadrzed:='@@@@@';
  end;
end;



(*
#################################################################
# Procedura sterujaca wybieraniem SZ z bazy                     #
#################################################################
*)
procedure CzytajSZzBazy;
var
  i: word;
begin
  Licz:=0;
  Nadrzed:='@@@@@';
  while (Licz<100) and
        (FilePos(ZbSrodZaklStary)<FileSize(ZbSrodZaklStary)) do
  begin
    i:=FilePos(ZbSrodZaklStary);
    Read(ZbSrodZaklStary, RekSrodZakl);
    if Nadrzed<>RekSrodZakl.Przynal then
    begin
```

```
    Licz:=Licz+1;
    RekJedn.Przynal:=RekSrodZakl.Przynal;
    RekJedn.NrRek:=i;
    WykJedn[Licz]:=RekJedn;
    Nadrzed:=RekSrodZakl.Przynal;
  end;  { if }
end;  { while }
PoczRek:=1;
Kontynuowac:='T';
while UpCase(Kontynuowac)<>'N' do
begin
  WybJednSZ(j);

  if UpCase(Kontynuowac)<>'N' then
  begin
    RekJedn:=WykJedn[j];
    i:=RekJedn.NrRek;
    Seek(ZbSrodZakl1Stary, i);
    Read(ZbSrodZakl1Stary, RekSrodZakl1Stary);
    Nadrzed:=RekSrodZakl1Stary.Przynal;

    GotoXY(25,21);
    HighVideo;
    Write('CZY CHCESZ POPRAWIAC (T/N) ?');
    NormVideo;
    ch:=ReadKey;
    if UpCase(ch)='T' then
    begin
      EdycjaSrodZakl;
      Kontynuowac:='T'
    end
    else
    begin
      if Ord(ch)=0 then ch:=ReadKey;
      ZapiszJednSZ
    end;  { else }
  end;  { if }
  end;  { while }
end;  { CzytajSZzBazy } .



(*
###################################################################
# Procedura  przydzielaj. i otwieraj. zbiory                      #
###################################################################
*)
procedure PrzydzZbSZ;
begin
  Assign(ZbSrodZakl, NazwaZb);
{$I-}
  Reset(ZbSrodZakl);
  if IOResult<>0 then Rewrite(ZBSrodZakl);
{$I+}
  Seek(ZbSrodZakl, FileSize(ZbSrodZakl));
  Assign(ZbSrodZakl1Stary, NazwaZbStarego);
{$I-}
  Reset(ZbSrodZakl1Stary);
  BrakBazy:=TRUE;
  if (IOResult=0) then
    if (FileSize(ZbSrodZakl1Stary)>0) then
      BrakBazy:=FALSE
```

```
    else
      Close(ZbSrodZaklStary);
{$I+}
end;  { PrzydzZbSZ }


(*
#################################################################
# Procedura sterujaca wprowadzaniem SZ z bazy i consoli   #
#################################################################
*)
procedure WprowSrodZakl;
begin
  PrzydzZbSZ;
  if (not BrakBazy) then CzytajSZzBazy;
  Maska;
  GotoXY(5,22);
  Write('CZY CHCESZ DOPISYWAC (T/N) ?');
  Kontynuowac:=ReadKey;
  if Ord(Kontynuowac)=0 then ch:=ReadKey;
  while UpCase(Kontynuowac)='T' do
  begin
    CzytajSZ(9);
    if UpCase(RekOK)='T' then Write(ZbSrodZakl, RekSrodZakl);
  end;
  Close(ZbSrodZakl);
  if (not BrakBazy) then Close(ZbSrodZaklStary);
end;  { WprowSrodZakl }


end.  { unit WRESCzyD }
```

Moduł WRE6Obl

```
unit WRE6Obl;

interface
uses
    WRE1Decl, Podst, Crt;
procedure Obliczenia;

implementation
const

     DystrCzasy: array[1..12] of integer=
                 (0, 10, 20, 30, 40, 50, 60,
                  120, 180, 240, 300, 1800);
     D: array[1..3, 1..12] of integer=
                 ((0, 168, 327, 492, 590, 654,
                  725, 962, 1077, 1173, 1263, 1922),
                  (0, 976, 1107, 1153, 1165, 1171,
                   1179, 1194, 1197, 1200, 1201, 1202),
                  (0, 1428, 1942, 2260, 2486, 2675,
                   2843, 3382, 3628, 3844, 3978, 4732));

     Kol: array[Zakr] of byte = (35, 45, 55);

var
   Odl: array[1..MaxWL, 1..MaxSZ] of integer;
   Bliskie: array[1..MaxWL, 1..MaxSZ] of byte;

   Czas, CzasTr, CzasZak, CzasRSyst: real;
   NrWez, IRE, KKK: byte;
   AS: real;
   WezZakl: array[1..MaxSZ] of byte;
   ZajSZ: array[1..MaxSZ] of real;
   PT: array[1..MaxWL] of real;
   CzasNZ, CzasTrZ: array[1..MaxZgl] of real;
   NrSZ, IREM: array[1..MaxZgl] of byte;
   ZakrZgl: array[1..MaxZgl] of Zakr;
   NrWL: array[1..MaxZgl] of byte;

   ZakrRel: Zakr;
   JestZgl: boolean;
   Ch: char;
   LR: byte;
   PNZ: byte;



(*
###############################################################
#  Procedura wprowadza dane o wezlach i srodkach zaklocen #
###############################################################
*)
procedure WprowadzDane;
var
   BladWeWy: boolean;
   i: byte;
begin
  WypDyr('Czytanie danych.','Prosze czekac!');
  Assign(ZbWezlow, 'WEZLACZ.OBL');
  Assign(ZbSrodZak, 'SRODZAK.OBL');
  BladWeWy:=FALSE;
```

```
{$I-}
  Reset(ZbWezlow);
  if IOResult<>0 then BladWeWy:=TRUE
  else
    begin
      Reset(ZbSrodZakl);
      if IOResult<>0 then BladWeWy:=TRUE;
    end;
{$I+}

  if not BladWeWy then
  begin
    LiczbaWezlow:=FileSize(ZbWezlow);
    LiczbaSrodZakl:=FileSize(ZbSrodZakl);
    AS:=0
  end;

  if LiczbaWezlow>MaxWL then BladWeWy:=TRUE;
  if LiczbaSrodZakl>MaxSZ then BladWeWy:=TRUE;

  if (LiczbaWezlow>0) and (not BladWeWy) then
  begin
    for i:=1 to LiczbaWezlow do
    begin
      Read(ZbWezlow, TabWL[i]);
      with TabWL[i] do
      begin
        LR:=LRU+LRK+LRR;
        if LR=0 then
        begin
          LRU:=1;
          LRK:=1;
          LRR:=1
        end
      end;
      AS:=AS+TabWL[i].A;
      PT[i]:=AS
    end;
    Close(ZbWezlow);
  end
  else
    BladWeWy:=TRUE;

  if (LiczbaSrodZakl>0) and (not BladWeWy) then
  begin
    for i:=1 to LiczbaSrodZakl do
    begin
      Read(ZbSrodZakl, TabSZ[i]);
      ZajSZ[i]:=0;
      WezZakl[i]:=0;
    end;
    Close(ZbSrodZakl);
  end
  else
    BladWeWy:=TRUE;

  if BladWeWy then
  begin
    GoToXY(5, 15);
    Write('Brak zbiorow danych do obliczen!');
    WypDyr('','Nacisnij klawisz!');
    Ch:=ReadKey;
```

```
    if Ord(Ch)=0 then Ch:=ReadKey;
    Kontynuowac:='N'
  end;  { if }

end;  { WprowadzDane }



(*
#############################################################
#  Procedura zestawia tablice odleglosci                    #
#############################################################
*)
procedure TabOdl;
var
   i, j, k, n: byte;
   rob, XW, YW, XSZ, YSZ: integer;

begin
  WypDyr('Tworzenie tabeli odleglosci.','Prosze czekac!');

  for i:=1 to LiczbaWezlow do
  begin
    for j:=1 to LiczbaSrodZakl do
    begin
      XW:=TabWL[i].X;
      YW:=TabWL[i].Y;
      XSZ:=TabSZ[j].X;
      YSZ:=TabSZ[j].Y;
      Odl[i,j]:=Round(Sqrt(Sqr((XW-XSZ)*1.0)+Sqr((YW-YSZ)*1.0)))
    end;  { for j }

    for k:=1 to LiczbaSrodZakl do
    begin
      rob:=10000;

      for j:=1 to LiczbaSrodZakl do
      begin
        if Odl[i,j]>=0 then
        begin
          if Odl[i,j]<rob then
          begin
            n:=j;
            rob:=Odl[i,j]
          end;
        end;
      end;  { for j }

      Bliskie[i,k]:=n;
      Odl[i,n]:=-Odl[i,n]-5;
    end;  { for k }

    for n:=1 to LiczbaSrodZakl do Odl[i,n]:=-Odl[i,n]-5;

  end;  { for i }

end;  { TabOdl }
```



```
(*
```

```
{*
#################################################################
#  Procedura generujaca czas nadejscia zgloszenia               #
#################################################################
*}
procedure GenerujCzasNadejscia;
var
   t: real;
begin
  t:=-Ln(Random);
  Czas:=Czas+(t/AS);

end;  { GenerujCzasNadejscia }


{*
#################################################################
#  Procedura generujaca zrodlo zgloszenia                       #
#################################################################
*}
procedure GenerujZrodlo;
var
   w: real;
   i, j: word;
   u, k, r: byte;
begin
  w:=Random*AS;
  NrWez:=1;
  while (w>PT[NrWez]) and (NrWez<LiczbaWezlow) do Inc(NrWez);
  u:=0;
  k:=0;
  r:=0;
  i:=1;
  while i<=KKK do
  begin
    if (NrWL[i]=NrWez) and (Czas<(CzasNZ[i]+CzasTrZ[i])) then
      case ZakrZgl[i] of
           UKF: Inc(u);
            KF: Inc(k);
            RL: Inc(r)
      end;
    Inc(i)
  end;
  with TabWL[NrWez] do
  begin
    u:=LRU-u;
    k:=LRK-k;
    r:=LRR-r;
  end;
  i:=u+k+r;
  if i>0 then
  begin
    j:=Random(i);
    if j<u then ZakrRel:=UKF else
      if j<u+k then ZakrRel:=KF else
        ZakrRel:=RL;
    JestZgl:=TRUE;
    Inc(LZg);
    Inc(TLZg[ZakrRel])
  end;

end;  { GenerujZrodlo }
```

```
{*
#################################################################
#  Procedura generujaca rodzaj emisji                           #
#################################################################
*}
procedure GenerujRodzajEmisji;
var
   w: word;
begin
  case ZakrRel of
      KF: begin
            w:=Random(D[1,12]+D[2,12]+D[3,12]);
            if w<D[1,12] then IRE:=1 else
              if w<(D[1,12]+D[2,12]) then IRE:=2 else
                IRE:=3;
          end;

     UKF: begin
            w:=Random(D[1,12]+D[3,12]);
            if w<D[1,12] then IRE:=1 else IRE:=3;
          end;

      RL: IRE:=1
  end
end;  { GenerujRodzajEmisji }


{*
#################################################################
#  Procedura generujaca czas trwania zgloszenia                 #
#################################################################
*}
procedure GenerujCzasTrwania;
var
   w: word;
   i: byte;
begin
  w:=Random(D[IRE,12]);
  i:=2;
  while w>=D[IRE,i] do Inc(i);
  CzasTr:=DystrCzasy[i-1]+
          ((DystrCzasy[i]-DystrCzasy[i-1])*
           ((w-D[IRE,i-1])/(D[IRE,i]-D[IRE,i-1])));
  CzasZak:=Czas+CzasTr;
end;  { GenerujCzasTrwania }


{*
#################################################################
#  Procedura generuje zgloszenie                                #
#################################################################
*}
procedure GenerujZgloszenie;
begin
  JestZgl:=FALSE;
  while not JestZgl do
  begin
    GenerujCzasNadejscia;
    GenerujZrodlo;
```

```
  end;
  GenerujRodzajEmisji;
  GenerujCzasTrwania;
end;  { GenerujZgloszenie }


(*
###############################################################
#  Procedura likwidujaca stare zgloszenia                     #
###############################################################
*)
procedure LikwStareZgloszenia;
var
   i,j: byte;
begin
  for i:=1 to LiczbaGrodZakl do WezZakl[i]:=0;

  i:=1;
  j:=1;

  while i<=KKK do
  begin
    if (CzasNZ[i]+CzasTrZ[i]>Czas) then
    begin
      if NrSZ[i]>0 then WezZakl[NrSZ[i]]:=NrWL[i];
      if (i>j) then
      begin
        CzasNZ[j]:=CzasNZ[i];
        CzasTrZ[j]:=CzasTrZ[i];
        NrSZ[j]:=NrSZ[i];
        IREM[j]:=IREM[i];
        ZakrZgl[j]:=ZakrZgl[i];
        NrWL[j]:=NrWL[i];
      end;
      j:=j+1;
    end;
    i:=i+1;
  end;

  if j<=MaxZgl then
  begin
    CzasNZ[j]:=Czas;
    CzasTrZ[j]:=CzasTr;
    IREM[j]:=IRE;
    ZakrZgl[j]:=ZakrRel;
    NrWL[j]:=NrWez;
    KKK:=j
  end
  else Kontynuowac:='N';
end;  { LikwStareZgloszenia }



(*
###############################################################
#  Procedury wyswietlajace wyniki symulacji na biezaco        #
###############################################################
*)

procedure OgLi;
begin
  GotoXY(Kol[ZakrRel], 6);
```

```
  Write(TLZg[ZakrRel]:10);
  GotoXY(Kol[ZakrRel], 10);
  Write((TLZg[ZakrRel]-TLiczZglZakl[ZakrRel]):10);
  GotoXY(Kol[ZakrRel], 18);
  Write((TLiczZglZakl[ZakrRel]/TLZg[ZakrRel]):10:3);
  GotoXY(65, 6);
  Write(LZg:10);
  GotoXY(65, 10);
  Write((LZg-LiczZglZakl):10);
  GotoXY(65, 18);
  Write((LiczZglZakl/LZg):10:3);
end;

procedure DrukNCzR;
begin
  OgLi;
  GotoXY(Kol[ZakrRel], 12);
  Write(TLiczNCzR[ZakrRel]:10);
  GotoXY(65, 12);
  Write(LiczNCzR:10);
end;

procedure DrukNWar;
begin
  OgLi;
  GotoXY(Kol[ZakrRel], 16);
  Write(TLiczNWar[ZakrRel]:10);
  GotoXY(65, 16);
  Write(LiczNWar:10);
end;

procedure DrukNZaj;
begin
  OgLi;
  GotoXY(Kol[ZakrRel], 14);
  Write(TLiczNZaj[ZakrRel]:10);
  GotoXY(65, 14);
  Write(LiczNZaj:10);
end;

procedure DrukZak;
begin
  OgLi;
  GotoXY(Kol[ZakrRel], 8);
  Write(TLiczZglZakl[ZakrRel]:10);
  GotoXY(65, 8);
  Write(LiczZglZakl:10);
end;


procedure DrukCzasILZg;
var
  Cz: longint;
  g, m, s: byte;
begin
  Cz:=Round(Czas);
  g:=Cz div 3600;
  m:=(Cz mod 3600) div 60;
  s:=Cz mod 60;
  GotoXY(11, 20);
  Write(g:2);
  GotoXY(20, 20);
```

```
  Write(m:2);
  GotoXY(28, 20);
  Write(s:2);
  GotoXY(70, 20);
  Write(KKK:5);
end;



{*
################################################################
#  Funkcja sprawdzajaca mozliwosc zaklocenia zgloszenia        #
################################################################
*}
function MozeZakl(var nast: boolean;
                  var CzRS: boolean;
                  var war: boolean;
                      SZak: byte): boolean;
var
   Rz, Rs, PZZ, PUU, KZZ: real;
   Mozl: boolean;
begin

                    {* zgodnosc zakresow *}
  if (Ord(ZakrRel)+1)=TabSZ[SZak].Zakres then Mozl:=TRUE
                                         else Mozl:=FALSE;

                    {* czas reakcji *}
  if Mozl then
  begin
    if (TabSZ[SZak].CzR)=CzasTr) then
      Mozl:=FALSE
    else
      CzRS:=TRUE
end;
                  {* warunki energetyczne *}
if Mozl then
begin
  Rz:=Sqr(Odl[NrWez, SZak]);
  with TabWL[NrWez] do
  begin
    Rs:=1.0*(DLM+Random(DMA-DLM+1));
    case ZakrRel of
      UKF: PUU:=Sqrt(PU);
       KF: PUU:=Sqrt(PK);
       RL: PUU:=Sqrt(PR)
    end
  end;

  with TabSZ[SZak] do
    PZZ:=Sqrt(P);

  if (PUU*Rz)<(5*PZZ*Rs) then
    KZZ:=5
  else
    KZZ:=(PZZ*Rs)/(PUU*Rz);

  if (ZakrRel=KF) or (ZakrRel=UKF) then
  begin
    case IRE of
      1, 3: if KZZ>=1 then Mozl:=TRUE else Mozl:=FALSE;
         2: if KZZ>=4 then Mozl:=TRUE else Mozl:=FALSE
```

```
      end;
      if Mozl then war:=TRUE;
    end
  end;

                  {* zajetosc stacji zaklocen *}
  if (WezZakl[ SZak] > 0) and
     (TabWL[NrWez].Prior >= TabWL[ WezZakl[ SZak]].Prior) then
    Mozl:=FALSE;


  MozeZakl:=Mozl;
end;  { MozeZakl }



{*
#############################################################
#  Procedura odnotowujaca zaklocenie lub niezaklocenie      #
#############################################################
*}
procedure ObsluzZgloszenie;
var
   nast, war, CzRS: boolean;
   SZak, i:byte;
begin
  i:=1;
  nast:=TRUE;
  war:=FALSE;
  CzRS:=FALSE;
  while nast and (i<=LiczbaSrodZakl) do
  begin
    SZak:=Bliskie[NrWez, i];

    if MozeZakl(nast, CzRS, war, SZak) then
      begin
        Inc(LiczZglZakl);
        Inc(TLiczZglZakl[ZakrRel]);
        ZajSZ[SZak]:=CzasZak;
        WezZakl[SZak]:=NrWez;
        nast:=FALSE
      end;

    i:=i+1;
  end;


  if nast then
  begin
    NrSZ[KKK]:=0;

    if not CzRS then
    begin
      NrSZ[KKK]:=0;
      Inc(LiczNCzR);
      Inc(TLiczNCzR[ZakrRel]);
      PNZ:=1;
      DrukNCzR;
    end

    else
    if (not war) and (ZakrRel<>RL) then
```

```
    begin
      Inc(LiczNWar);
      Inc(TLiczNWar[ZakrRel]);
      PNZ:=2;
      DrukNWar
    end

    else
    begin
      Inc(TLiczNZaj[ZakrRel]);
      Inc(LiczNZaj);
      PNZ:=3;
      DrukNZaj
    end;
  end

  else
  begin
    NrSZ[KKK]:=SZak;
    PNZ:=0;
    DrukZak
  end;


  with RekZgl do
  begin
    CzNZ:=Czas;
    CzTr:=CzasTr;
    NWL:=NrWez;
    NSZ:=NrSZ[KKK];
    REM:=IRE;
    ZZgl:=ZakrRel;
    PowodNZ:=PNZ;
  end;
  Write(ZbZgloszen, RekZgl);

  DrukCzasILZg
end;  { ObsluzZgloszenie }



(*
#################################################################
#  Procedura drukuje wyniki                                     #
#################################################################
*)
procedure EkranWynikow;
begin
  Maska;
  WypDyr('Prosze czekac!','Jesli chcesz przerwac nacisnij klawisz!');
  GoToXY(35, 4);
  Write('       UKF        KF     R/lin     Razem');
  GoToXY(5, 6);
  Write('     Ogolna liczba zgloszen: ');
  GoToXY(5, 8);
  Write('                  Zakloconych: ');
  GoToXY(5, 10);
  Write(' Niezakloconych:               ');
  GoToXY(5, 12);
  Write('-  ze wzgl. na czas reakcji: ');
  GoToXY(5, 14);
```

```
  Write('-  ze wzgl. na zaj. systemu: ');
  GoToXY(5, 16);
  Write('- ze wzgl. na warunki (moc): ');
  GoToXY(5, 18);
  Write('       Efektywnosc zaklocen: ');
  GoToXY(5, 20);
  Write('Czas:    godz.    min.    sek.');
  GoToXY(46, 20);
  Write('Liczba zgloszen w toku:');

end;  { EkranWynikow }



(*
#################################################################
#  Program sterujacy obliczeniami                               #
#################################################################
*)
procedure Obliczenia;
var
   nrzg: string[6];
begin
  Maska;
  Kontynuowac:='T';
  WprowadzDane;
  if UpCase(Kontynuowac)<>'N' then
  begin
    TabOdl;
    Randomize;
    LiczZglZakl:=0;
    Czas:=0;
    LZg:=0;
    LiczZglZakl:=0;
    LiczNWar:=0;
    LiczNZaj:=0;
    LiczNCzR:=0;
    for ZakrRel:=UKF to RL do
    begin
      TLZg[ZakrRel]:=0;
      TLiczZglZakl[ZakrRel]:=0;
      TLiczNWar[ZakrRel]:=0;
      TLiczNZaj[ZakrRel]:=0;
      TLiczNCzR[ZakrRel]:=0
    end;
    KKK:=0;

    Assign(ZbZgloszen, 'ZGLOSZEN.OBL');
    Rewrite(ZbZgloszen);

    EkranWynikow;

    while (UpCase(Kontynuowac)<>'N') and (LZg<1000) do
    begin
      GenerujZgloszenie;
      LikwStareZgloszenia;
      if Kontynuowac<>'N' then
      begin
        ObsluzZgloszenie;
        if KeyPressed then
        begin
```

```
            Kontynuowac:='N';
            Ch:=ReadKey;
            if Ord(Ch)=0 then Ch:=ReadKey;
          end
        end
        else
        begin
          Str(MaxZgl:4,nrzg);
          WypDyr('Za duzo zgloszen w toku - ponad '+nrzg,
                 'Nacisnij klawisz!');
          Ch:=ReadKey;
          if Ord(Ch)=0 then Ch:=ReadKey;
        end
      end;
      WypDyr('','Nacisnij klawisz!');
      Ch:=ReadKey;
      if Ord(Ch)=0 then Ch:=ReadKey;
      Close(ZbZgloszen);
      Kontynuowac:='T';
    end;
  end;  { Obliczenia }

end.
```

Moduł WRE7WypW

```
unit WRE7WypW;

interface
uses
  WRE1Decl, Podst, Crt;
procedure WyswWez;

implementation
type
   WynWez = record
                ZgSzt: word;
                Czas:  real;
                Zakl:  word;
                NCz, NW, NZaj: word;
                PZgl: byte;
                PCz, SrOp, CzasZN: real
            end;

var
   WybraneWL: array[1..15] of byte;
   LWyb,PoczRek: byte;

   OgWynWez: array[0..15] of WynWez;
   SzWynWez: array[Zakr, 0..15] of WynWez;

const
  WW: array[0..3] of byte = (4, 9, 14, 19);
  Text: array[Zakr] of string[13]=
           ('w tym:   UKF',
            '          KF',
            '         R/1');


{$
#############################################################
# Procedura drukujaca dla proc. wybierania WL               #
#############################################################
$}
procedure DrWybWL(i: byte);
begin
  GotoXY(5, 3+i);
  case i of
       1: Write(' ... poprzednie wezly');
    2..15: if (PoczRek+i-2)>LiczbaWezlow then
              Write('                    ')
           else
           begin
             Write(TabWL[PoczRek+i-2].Przynal);
             GotoXY(16, 3+i);
             Write(TabWL[PoczRek+i-2].Nazwa);
           end;
      16: Writeln('nastepne wezly ...');
      17: Writeln('KONIEC WYBIERANIA WEZLOW')
  end;  { case }
end;  { DrWybWL }


{$
#############################################################
```

```
# Procedura wybierania jednego Wez. Lacz.                #
##########################################################
*)
procedure WybiWL(var j: byte);
var
  i: byte;
  Druk: WWiersz;
begin
  Druk:=DrWybWL;
  i:=1;
  while (i=1) or (i=16) or
        ((i<17) and ((PoczRek+i-2)>LiczbaWezlow)) do
  begin
    i:=Wybor(@@Druk, 17, FALSE);
    case i of
         1: if (PoczRek-14)>0 then PoczRek:=PoczRek-14
            else PoczRek:=1;
     2..15: j:=PoczRek+i-2;
        16: if (PoczRek+14)>LiczbaWezlow then PoczRek:=LiczbaWezlow
            else PoczRek:=PoczRek+14;
        17: Kontynuowac:='N'
    end;  { case }
  end;  { while }
end;  { WybWL }


(*
##########################################################
# Procedura wybierania Wezlow Lacznosci                  #
##########################################################
*)
procedure WybWL;
var
  jest: boolean;
  i, j: byte;
begin
  PoczRek:=1;
  for i:=1 to LiczbaWezlow do WybraneWL[i]:=0;
  Maska;
  WypDyr(kom1, kom2);
  GotoXY(40, 3);
  Write('WEZLY WYBRANE:');
  LWyb:=0;
  Kontynuowac:='T';
  while (Kontynuowac<>'N') and (LWyb<15) do
  begin
    WybiWL(j);
    if Kontynuowac<>'N' then
    begin
      jest:=FALSE;
      i:=1;
      while (i<=LWyb) and not jest do
      begin
        if WybraneWL[i]=j then jest:=TRUE;
        Inc(i)
      end;
      if jest then
        Beep
      else
      begin
        Inc(LWyb);
```

```
        WybraneWL[LWyb]:=j;
        GotoXY(40, 3+LWyb);
        Write(TabWL[j].Przynal);
        GotoXY(51, 3+LWyb);
        Write(TabWL[j].Nazwa)
      end
    end
  end;
end;  { WybWL }



(*
###################################################################
# Procedura obliczajaca wyniki dla wybranej grupy Wez. L. #
###################################################################
*)
procedure OblWez;
var
  i, j: word;
  k: Zakr;
  jest: boolean;
begin

  for i:=0 to 15 do
  begin
    with OgWynWez[i] do
    begin
      ZgSzt:=0;
      Czas:=0;
      Zakl:=0;
      Ncz:=0;
      Nw:=0;
      NZaj:=0;
      PZgl:=0;
      PCz:=0;
      SrOp:=0;
      CzasZN:=0;
    end;

    for k:= UKF to RL do
    with SzWynWez[k, i] do
    begin
      ZgSzt:=0;
      Czas:=0;
      Zakl:=0;
      Ncz:=0;
      Nw:=0;
      NZaj:=0;
      PZgl:=0;
      PCz:=0;
      SrOp:=0;
      CzasZN:=0;
    end;
  end;

  Reset(ZbZgloszen);
  LZg:=FileSize(ZbZgloszen);

  for i:=1 to LZg do
  begin
    Read(ZbZgloszen, RekZgl);
```

```
jest:=FALSE;
j:=1;
while (j<=LWyb) and not jest do
  if WybraneWL[j]=RekZgl.NWL then
    jest:=TRUE
  else
    Inc(j);

if jest then
begin
  with OgWynWez[j] do
  begin
    Inc(ZgSzt);
    Czas:=Czas+RekZgl.CzTr;
    if (RekZgl.NSZ<>0) then
    begin
      Inc(Zakl);
      Inc(PZgl);
      PCz:=PCz+RekZgl.CzTr;
      if CzasZN=0 then CzasZN:=RekZgl.CzNZ;
    end
    else
    begin
      case RekZgl.PowodNZ of
         1: Inc(NCz);
         2: Inc(NW);
         3: Inc(NZaj);
      end;
      if CzasZN<>0 then
      begin
        SrOp:=SrOp+(RekZgl.CzNZ-CzasZN);
        CzasZN:=0;
      end;
    end;
  end;  { with }

k:=RekZgl.ZZgl;
with SzWynWez[k, j] do
begin
  Inc(ZgSzt);
  Czas:=Czas+RekZgl.CzTr;
  if (RekZgl.NSZ<>0) then
  begin
    Inc(Zakl);
    Inc(PZgl);
    PCz:=PCz+RekZgl.CzTr;
    if CzasZN=0 then CzasZN:=RekZgl.CzNZ;
  end
  else
  begin
    case RekZgl.PowodNZ of
       1: Inc(NCz);
       2: Inc(NW);
       3: Inc(NZaj);
    end;
    if CzasZN<>0 then
    begin
      SrOp:=SrOp+(RekZgl.CzNZ-CzasZN);
      CzasZN:=0;
    end;
  end;
end;  { with }
```

```
      with OgWynWez[0] do
      begin
        Inc(ZgSzt);
        Czas:=Czas+RekZgl.CzTr;
        if (RekZgl.NSZ<>0) then
        begin
          Inc(Zakl);
          Inc(PZgl);
          PCz:=PCz+RekZgl.CzTr;
          if CzasZN=0 then CzasZN:=RekZgl.CzNZ;
        end
        else
        begin
          case RekZgl.PowodNZ of
             1: Inc(NCz);
             2: Inc(NW);
             3: Inc(NZaj);
          end;
          if CzasZN<>0 then
          begin
            SrOp:=SrOp+(RekZgl.CzNZ-CzasZN);
            CzasZN:=0;
          end;
        end;
      end;  { with }

      with SzWynWez[k, 0] do
      begin
        Inc(ZgSzt);
        Czas:=Czas+RekZgl.CzTr;
        if (RekZgl.NSZ<>0) then
        begin
          Inc(Zakl);
          Inc(PZgl);
          PCz:=PCz+RekZgl.CzTr;
          if CzasZN=0 then CzasZN:=RekZgl.CzNZ;
        end
        else
        begin
          case RekZgl.PowodNZ of
             1: Inc(NCz);
             2: Inc(NW);
             3: Inc(NZaj);
          end;
          if CzasZN<>0 then
          begin
            SrOp:=SrOp+(RekZgl.CzNZ-CzasZN);
            CzasZN:=0;
          end;
        end;
      end;  { with }

    end;  { if jest }
  end;  { for i:=1 to LZg Read... }


  for i:=0 to LWyb do
  begin
    with OgWynWez[i] do
    begin
      if CzasZN<>0 then
```

```
          SrOp:=SrOp+(RekZgl.CzNZ-CzasZN);
        if ZgSzt>0 then
        begin
          PZgl:=Round((PZgl/ZgSzt)*100);
          SrOp:=SrOp/ZgSzt
        end;
        if Czas>0.1 then
          PCz:=(PCz/Czas)*100;
      end;

      for k:=UKF to RL do
      with SzWynWez[k, i] do
      begin
        if CzasZN<>0 then
          SrOp:=SrOp+(RekZgl.CzNZ-CzasZN);
        if ZgSzt>0 then
        begin
          PZgl:=Round((PZgl/ZgSzt)*100);
          SrOp:=SrOp/ZgSzt
        end;
        if Czas>0.1 then
          PCz:=(PCz/Czas)*100;
      end;
    end;  { for i:=1 to LWyb }


end;  { OblWez }
```

```
(*
#####################################################################
# Procedura wyswietlajaca wyniki dla 4 wezlow                       #
#####################################################################
*)
procedure Wysw4Wez(Pocz: byte);
var
  i, j: byte;
  k: Zakr;
begin
  Maska;
  WypDyr('','Nacisnij klawisz!');
  GotoXY(8, 3);
  Write('WEZLY');
  GotoXY(24, 2);
  Write(' ZGLOSZENIA  ZAKLOC.  NIEZAKLOCONE  '+
        'EFEKTYWNOSC ZAKLOC.');
  GotoXY(24, 3);
  Write(' Szt    Czas   Szt    CzR  War ZajS '+
        '%Zgl %Czasu SrOpozn.');

  for i:=0 to 3 do
  if (Pocz+i<=LWyb) then
  begin
    begin
      GoToXY(2, WW[i]);
      with TabWL[WybraneWL[Pocz+i]] do
        Write(Przynal+' '+Nazwa);
      GotoXY(24, WW[i]);
      with OgWynWez[Pocz+i] do
```

```
        Write(ZgSzt:4, Czas:8:0, Zakl:6, NCz:6, NW:5, NZaj:5,
              PZgl:6, PCz:6:0, SrOp:8:0, '"');
      for k:=UKF to RL do
      begin
        GotoXY(10, WW[i]+Ord(k)+1);
        with SzWynWez[k, Pocz+i] do
        Write(Text[k],
              ZgSzt:5, Czas:8:0, Zakl:6, NCz:6, NW:5,
               NZaj:5, PZgl:6, PCz:6:0, SrOp:8:0, '"');
      end;
    end;
  end;  { for i }

end;


(*
#############################################################
# Procedura wyswietlajaca wyniki gobalne dla grupy Wez.     #
#############################################################
*)
procedure WyswTotal;
var
  i: byte;
  k: Zakr;
const
  Text1: array[Zakr] of string[9]=
           ('w tym:UKF',
            '      KF',
            '     R/1');

begin
  Maska;
  WypDyr('','Nacisnij klawisz!');
  GotoXY(13, 4);
  Write(' ZGLOSZENIA  ZAKLOC.  NIEZAKLOCONE   '+
        'EFEKTYWNOSC ZAKLOC.');
  GotoXY(13, 5);
  Write(' Szt    Czas   Szt   CzR  War ZajS  '+
        '%Zgl %Czasu SrOpozn.');

  GotoXY(5, 7);
  with OgWynWez[0] do
    Write(' Razem: ',
          ZgSzt:4, Czas:8:0, Zakl:6, NCz:6, NW:5, NZaj:5,
          PZgl:6, PCz:6:0, SrOp:8:0, '"');

  for k:=UKF to RL do
  begin
    GotoXY(3, 9+Ord(k));
    with SzWynWez[k, 0] do
    Write(Text1[k],
          ZgSzt:5, Czas:8:0, Zakl:6, NCz:6, NW:5,
          NZaj:5, PZgl:6, PCz:6:0, SrOp:8:0, '"');
  end;

  GotoXY(3, 15);
  Write('WYBRANA GRUPA WEZLÓW:');
  for i:=1 to LWyb do
  begin
    GoToXY(10+20*((i-1) mod 3), Trunc(16+(i-1)/3));
```

```
    with TabWL[WybraneWL[i]] do
      Write(Przynal+' '+Nazwa);
  end;

end;  { WyswTotal }




(*
#############################################################
# Procedura wyswietlania danych o Wezlach Lacznosci         #
#############################################################
*)
procedure WyswWez;
var
  i: byte;
  Ch: char;
begin
  Kontynuowac:='T';
  while UpCase(Kontynuowac)<>'N' do
  begin
    WypDyr('','');
    GotoXY(20, 23);
    HighVideo;
    Write('Chcesz zestawiac grupe wezlow (T/N)?');
    NormVideo;
    Kontynuowac:=ReadKey;
    if Ord(Kontynuowac)=0 then Ch:=ReadKey;
    if UpCase(Kontynuowac)<>'N' then
    begin
      WybWL;
      OblWez;
      if LWyb>0 then
      begin
        i:=1;
        while i<=LWyb do
        begin
          Wysw4Wez(i);
          Ch:=ReadKey;
          if Ord(Ch)=0 then Ch:=ReadKey;
          Inc(i,4)
        end;  { while }
        WyswTotal;
        Ch:=ReadKey;
        if Ord(Ch)=0 then Ch:=ReadKey;
      end;  { if }
      Kontynuowac:='T';
    end;  { if }
  end;  { while }
end;  { WyswWez }

end.  { WRE7WypW }
```

Moduł WRE

```
program WRE;

uses
    WRE7WypW, WRE6Obl, WRE5CzyD, WRE4WypB,
    WRE3AktB, WRE2ZakB, WRE1Decl, Podst, Crt;

var
  c: char;



{$
#############################################################################
#  Procedura wyprowadzania wiersza oferty dla procedury "WprowTab"/Wybor/ #
#              Dla NIEBIESKICH / Dla CZERWONYCH                             #
#############################################################################
$}
procedure WWprowTab1(NrWier: byte);
  begin
     GotoXY(50, 10+NrWier);
     case NrWier of
          1: Write(' Dla NIEBIESKICH ');
          2: Write(' Dla CZERWONYCH  ');

     end
  end;


{$
#############################################################################
#  Procedura wyprowadzania wiersza oferty dla procedury "WprowTab"/Wybor/ #
#          Wezly lacznosci / Srodki zaklocajace                             #
#############################################################################
*)
procedure WWprowTab2(NrWier: byte);
  begin
     GotoXY(25,10+NrWier);
     case NrWier of
          1: Write(' Wezly lacznosci    ');
          2: Write(' Srodki zaklocajace ');
     end
  end;


{$
#############################################################################
#  Procedura wyprowadzania wiersza oferty dla procedury "WprowTab"/Wybor/ #
#          Dopisywanie / Poprawianie                                        #
#############################################################################
$}
procedure WWprowTab3(NrWier: byte);
  begin
     GotoXY(5, 10+NrWier);
     case NrWier of
          1: Write(' Dopisywanie  ');
          2: Write(' Poprawianie  ');
          3: Write(' Wyswietlanie ');
          4: Write(' Drukowanie   ');

     end
  end;
```

```
(*
#############################################################################
#  Procedura okreslania Strony, Obiektow i Czynnosci                        #
#############################################################################
*)
procedure OkreslZadanie;
var
   wariant: integer;
   Druk: WWiersz;
begin
  Druk:=WWprowTab3;
  wariant:=Wybor(@@Druk,4,FALSE);
  case wariant of
       1: Dzialanie:=Dopisywanie;
       2: Dzialanie:=Poprawianie;
       3: Dzialanie:=Wyswietlanie;
       4: Dzialanie:=Drukowanie
  end;  { case }

  Druk:=WWprowTab2;
  wariant:=Wybor(@@Druk,2,FALSE);
  case wariant of
       1: Obiekty:=WezlyLacz;
       2: Obiekty:=SrodkiZakl
  end;  { case }

  if (c='B') or (Dzialanie=Dopisywanie) then
  begin
    Druk:=WWprowTab1;
    wariant:=Wybor(@@Druk,2,FALSE);
    case wariant of
         1: Strona:=Niebiescy;
         2: Strona:=Czerwoni
    end;  { case }
  end;  { if }

end;  { OkreslZadanie }



(*
#############################################################################
# Procedura przedstawia oferte wprowadzania danych stalych   jednej         #
# ze stron i wywoluje procedure realizujaca wybrana funkcje.                #
#############################################################################
*)
procedure WprowTab;
begin
  c:='B';
  OkreslZadanie;
  case Strona of
      Niebiescy:
        case Obiekty of
            WezlyLacz:
               begin
                 NazwaZb:='WEZLACZ.NIE';
                 NazwaZbStarego:='WEZLACZ.NNN';
               end;
            SrodkiZakl:
```

```
            begin
              NazwaZb:='SRODZAK.NIE';
              NazwaZbStarego:='SRODZAK.NNN';
            end;
      end;
    Czerwoni:
      case Obiekty of
          WezlyLacz:
            begin
              NazwaZb:='WEZLACZ.CZE';
              NazwaZbStarego:='WEZLACZ.CCC';
            end;
          SrodkiZakl:
            begin
              NazwaZb:='SRODZAK.CZE';
              NazwaZbStarego:='SRODZAK.CCC';
            end
      end
end;

case Obiekty of
    WezlyLacz:
      case Dzialanie of
          Dopisywanie: DopWL;
          Poprawianie: PopWL;
          Wyswietlanie: WyswWL;
          Drukowanie: DrukWL
      end;
    SrodkiZakl:
      case Dzialanie of
          Dopisywanie: DopSZ;
          Poprawianie: PopSZ;
          Wyswietlanie: WyswSZ;
          Drukowanie: DrukSZ
        end
  end;

end;



(*
##############################################################################
# Procedura przedstawia oferte wprowadzania danych zmiennych jednej          #
# ze stron i wywoluje procedure realizujaca wybrana funkcje.                 #
##############################################################################
*)
procedure WprowZmiennych;
begin
  Strona:=Niebiescy;
  c:='Z';
  OkreslZadanie;
  case Strona of
      Niebiescy:
        case Obiekty of
            WezlyLacz:
              begin
                NazwaZb:='WEZLACZ.OBL';
                NazwaZbStarego:='WEZLACZ.NIE';
              end;  { WezlyLacz }
            SrodkiZakl:
              begin
```

```
                NazwaZb:='SRODZAK.OBL';
                NazwaZbStarego:='SRODZAK.NIE';
              end;  { SrodkiZakl }
        end;  { Obiekty }

    Czerwoni:
       case Obiekty of
           WezlyLacz:
              begin
                NazwaZb:='WEZLACZ.OBL';
                NazwaZbStarego:='WEZLACZ.CZE';
              end;  { WezlyLacz }
           SrodkiZakl:
              begin
                NazwaZb:='SRODZAK.OBL';
                NazwaZbStarego:='SRODZAK.CZE';
              end  { SrodkiZakl }
       end  { case Obiekty }
end;  { case Strona }

case Obiekty of
    WezlyLacz:
       case Dzialanie of
           Dopisywanie: WprowWezLacz;
           Poprawianie:
              begin
                NazwaZbStarego:='WEZLACZ.BAK';
                PopWL
              end;
           Wyswietlanie: WyswWL;
           Drukowanie: DrukWL
       end;  { case }
      SrodkiZakl:
         case Dzialanie of
             Dopisywanie: WprowSrodZakl;
             Poprawianie:
                begin
                  NazwaZbStarego:='SRODZAK.BAK';
                  PopSZ
                end;  { Poprawianie }
             Wyswietlanie: WyswSZ;
             Drukowanie: DrukSZ
         end  { case Dzialanie }
  end
end;  { WprowZmiennych }
```

```
(*
#############################################################################
#  Procedura druku wiersza dla programu /Wybor/                             #
#############################################################################
*)
procedure WProg(NrWiersza: byte);
begin
  GotoXY(5,5+NrWiersza);
  case NrWiersza of
       1: Write( '1. Obliczenia /na zestawionych danych zmiennych/');
       2: Write( '2. Zestawianie danych zmiennych');
```

```
      3: Write( '3. Zestawianie danych stalych');
      4: Write( '4: Zakonczenie dzialania')
  end
end;


(*
#############################################################################
#   P  R  O  G  R  A  M                                                     #
#############################################################################
*)

var
   wariant: integer;
   Druk: WWiersz;

begin
   Druk:=WProg;
   wariant:=0;
   while wariant < 4 do
     begin
        wariant:=Wybor(@@Druk,4,TRUE);
        case wariant of
             1: begin
                  Obliczenia;
                  if Kontynuowac<>'N' then WyswWez;
                end;
             2: WprowZmiennych;
             3: WprowTab;
        end
     end
end.
```

MASKI OFERT PROGRAMU

Załącznik 4

maska 1

KSO ASG WP

1. Obliczenia /na zestawionych danych zmiennych/
2. Zestawianie danych zmiennych
3. Zestawianie danych stalych
4. Zakonczenie dzialania

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

maska 2

KSO ASG WP

1. Obliczenia /na zestawionych danych zmiennych/
2. Zestawianie danych zmiennych
3. Zestawianie danych stalych
4. Zakonczenie dzialania

Dopisywanie
Poprawianie
Wyswietlanie
Drukowanie

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

maska 3

KSO ASG WF

1. Obliczenia /na zestawionych danych zmiennych/
2. Zestawianie danych zmiennych
3. Zestawianie danych stałych
4. Zakończenie działania

Dopisywanie
Poprawianie
Wyswietlanie
Drukowanie

Wezły łączności
Środki zakłocające

Wybierz używając klawiszy: Home, End, PgUp, PgDn.
Po wybraniu naciśnij Enter.

maska 4

KSO ASG WF

1. Obliczenia /na zestawionych danych zmiennych/
2. Zestawianie danych zmiennych
3. Zestawianie danych stałych
4. Zakończenie działania

Dopisywanie
Poprawianie
Wyswietlanie
Drukowanie

Wezły łączności
Środki zakłocające

Dla NIEBIESKICH
Dla CZERWONYCH

Wybierz używając klawiszy: Home, End, PgUp, PgDn.
Po wybraniu naciśnij Enter.

maska 5

KSD ASG WP

```
          Nazwa jednostki (maks. 10 znakow):        2KA
  Nazwa wezla lacznosci (maks. 10 znakow):          4DZ
      Wspolrzedna X wezla lacznosci: X =             70
      Wspolrzedna Y wezla lacznosci: Y =             20
    Priorytet wezla lacznosci: Prior =                3
  Liczba relacji KF wezla lacznosci: LRK =            2
 Liczba relacji UKF wezla lacznosci: LRU =            2
Liczba relacji radiol. wezla lacznosci: LRR =         1
  Minimalna dlugosc linii lacznosci: DLM =           20
 Maksymalna dlugosc linii lacznosci: DMA =           80
  Srednie natezenie ruchu erlangach:   A =        0.200
          Moc promieniowania KF: PK =              100
         Moc promieniowania UKF: PU =              100
    Moc promieniowania radiolinii: PR =            100
          ZAPISAC DO ZBIORU (T/N) ? :                T
CZY WPROWADZASZ NASTEPNY WEZEL (T/N) ? :  N
```

Dopisywanię danych o wezlach lacznosci NIEBIESKICH
Kazda dana koncz nacisnieciem klawisza Enter.

maska 6

KSD ASG WP

```
          Nazwa jednostki (maks. 10 znakow):        1A
  Nazwa srodka zaklocen (maks. 10 znakow):          SZ-5
      Wspolrzedna X srodka zaklocen: X =             50
      Wspolrzedna Y srodka zaklocen: Y =             80
          Moc srodka zaklocen: P =                  300
 Czas reakcji srodka zaklocen (sek.): CzR =          15
          Zakres (1-UKF, 2-KF, 3-R/1) :             KF
          ZAPISAC DO ZBIORU (T/N) ? :                T
CZY WPROWADZASZ NASTEPNY SROD. ZAKL.(T/N) ? :  N
```

Wprowadzanie danych o srodkach zaklocen CZERWONYCH
Kazda dana koncz nacisnieciem klawisza Enter.

KSO ASG WP

maska 7

Wezly lacznosci: 1KA

Zachowanie wezlow lacznosci danej jednostki
Skasowanie wezlow lacznosci danej jednostki
Poprawianie wezlow lacznosci danej jednostki
Zachowanie wezlow lacznosci reszty jednostek
Skasowanie wezlow lacznosci reszty jednostek

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

KSO ASG WP

maska 8

Nazwa jednostki (maks. 10 znakow):	1KA
Nazwa wezla lacznosci (maks. 10 znakow):	SD
Wspolrzedna X wezla lacznosci: X =	50
Wspolrzedna Y wezla lacznosci: Y =	50
Priorytet wezla lacznosci: Prior =	1
Liczba relacji KF wezla lacznosci: LRK =	3
Liczba relacji UKF wezla lacznosci: LRU =	3
Liczba relacji radiol. wezla lacznosci: LRR =	2
Minimalna dlugosc linii lacznosci: DLM =	20
Maksymalna dlugosc linii lacznosci: DMA =	180
Srednie natezenie ruchu erlangach: A =	0.350
Moc promieniowania KF: PK =	200
Moc promieniowania UKF: PU =	100
Moc promieniowania radiolinii: PR =	100

ZACHOWAC SKASOWAC ZMIENIAC DOPISAC PRZED

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

maska 9

KSO ASG WP

Nazwa jednostki (maks. 10 znakow):	1KA
Nazwa wezla lacznosci (maks. 10 znakow):	SD
Wspolrzedna X wezla lacznosci: X =	30
Wspolrzedna Y wezla lacznosci: Y =	50
Priorytet wezla lacznosci: Prior =	1
Liczba relacji KF wezla lacznosci: LRK =	3
Liczba relacji UKF wezla lacznosci: LRU =	3
Liczba relacji radiol. wezla lacznosci: LRR =	2
Minimalna dlugosc linii lacznosci: DLM =	20
Maksymalna dlugosc linii lacznosci: DMA =	180
Srednie natezenie ruchu erlangach: A =	0.350
Moc promieniowania KF: PK =	200
Moc promieniowania UKF: PU =	100
Moc promieniowania radiolinii: PR =	100
ZAPISAC DO ZBIORU (T/N) ? :	T
CZY ZAKONCZYLES EDYCJE WEZLA (T/N) ?	N

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Nastepnie nacisnij Enter, wprowadz wartosc i nacisnij Enter

maska 10

KSO ASG WP

Srodki zaklocen jednostki: 1A.

Zachowanie srodkow zaklocen danej jednostki
Skasowanie srodkow zaklocen danej jednostki
Poprawianie srodkow zaklocen danej jednostki
Zachowanie srodkow zaklocen reszty jednostek
Skasowanie srodkow zaklocen reszty jednostek

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

maska 11

KSO AS6 WP

Nazwa jednostki (maks. 10 znakow):	1A
Nazwa srodka zaklocen (maks. 10 znakow):	SZ-1
Wspolrzedna X srodka zaklocen: X =	20
Wspolrzedna Y srodka zaklocen: Y =	90
Moc srodka zaklocen: P =	200
Czas reakcji srodka zaklocen (sek.): CzR =	2
Zakres (1-UKF, 2-KF, 3-R/1) :	UKF

ZACHOWAC SKASOWAC ZMIENIAC DOPISAC PRZED

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

maska 12

KSO AS6 WP

Nazwa jednostki (maks. 10 znakow):	1A
Nazwa srodka zaklocen (maks. 10 znakow):	SZ-1
Wspolrzedna X srodka zaklocen: X =	20
Wspolrzedna Y srodka zaklocen: Y =	90
Moc srodka zaklocen: P =	200
Czas reakcji srodka zaklocen (sek.): CzR =	2
Zakres (1-UKF, 2-KF, 3-R/1) :	UKF
ZAPISAC DO ZBIORU (T/N) ? :	T
CZY ZAKONCZYLES EDYCJE SRODKA ZAKL. (T/N) ? :	N

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Nastepnie nacisnij Enter, wprowadz wartosc i nacisnij Enter

maska 13

KSD ASG WP

Jednostka	Wezel	X	Y	Pr	LK	LU	LR	DLM	DMA	A	PK	PU	PR
1KA	SD	50	50	1	3	3	2	20	180	0.350	200	100	100
1KA	1DZ	20	80	2	2	2	1	30	100	0.250	100	100	100
1KA	2DZ	50	80	2	2	2	1	30	100	0.250	200	200	100
1KA	3DZ	80	80	2	2	2	1	30	100	0.250	150	150	100
1KA	4DZ	50	30	3	2	2	1	20	50	0.200	150	150	100
2KA	SD	50	50	1	3	3	2	20	150	0.500	200	200	200
2KA	1DZ	35	80	2	2	2	1	20	80	0.300	100	100	100
2KA	2DZ	70	80	2	2	2	1	20	80	0.250	100	100	100
2KA	3DZ	50	30	3	2	2	1	20	60	0.200	100	100	100
2KA	4DZ	70	20	3	2	2	1	20	80	0.200	100	100	100
2KA	4DZ	70	20	3	2	2	1	20	80	0.200	100	100	100

Nacisnij klawisz, Esc - koniec wyswietlania.

maska 14

KSD ASG WP

Jednostka	Sr. zakl.	X	Y	Moc	Cz.r.	Zakr.
1A	SZ-1	20	90	200	2	UKF
1A	SZ-3	50	90	300	4	R/1
1A	SZ-4	60	90	400	2	R/1
1A	SZ-5	65	90	300	7	UKF
1A	SZ-6	70	90	200	2	KF
2A	SZ-1	20	90	500	3	UKF
2A	SZ-2	40	90	300	3	KF
2A	SZ-3	50	90	200	4	R/1
2A	SZ-4	60	90	300	5	UKF
2A	SZ-5	70	90	400	2	KF

Nacisnij klawisz, Esc - koniec wyswietlania.

maska 15

KSO ASG WP

... poprzednie jednostki
1KA
2KA

nastepne jednostki ...
KONIEC WYBIERANIA Z BAZY

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

maska 16

KSO ASG WP

... poprzednie jednostki
1KA
2KA

nastepne jednostki ...
KONIEC WYBIERANIA Z BAZY
CZY CHCESZ POPRAWIAC (T/N) ?
Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

maska 17

KSO ASG WP

Nazwa jednostki (bez zmian):

Wspolrzedne lewego skrzydla frontu jednostki:
Xfl= 0   Yfl= 10

Wspolrzedne prawego skrzydla frontu jednostki:
Xfp= 10   Yfp= 10

Wspolrzedne lewego skrzydla tylu jednostki:
Xtl= 0   Ytl= 0

Wspolrzedne prawego skrzydla tylu jednostki:
Xtp= 10   Ytp=0

Wprowadz podane wielkosci
Kazda koncz Enter

maska 18

KSO ASG WP

CZY CHCESZ DOPISYWAC (T/N) ?

Załącznik 5

WYNIKI PROGRAMU

maska 1

KSO ASG WP

	UKF	KF	R/lin	Razem
Ogolna liczba zgloszen:	444	391	165	1000
Zakloconych:	77	23	32	132
Niezakloconych:	367	368	133	868
- ze wzgl. na czas reakcji:	144	140	45	329
- ze wzgl. na zaj. systemu:	132	43	88	263
- ze wzgl. na warunki (moc):	91	185		276
Efektywnosc zaklocen:	0.173	0.059	0.194	0.132

Czas: 1 godz. 28 min. 2 sek. Liczba zgloszen w toku: 51

Nacisnij klawisz!

maska 2

KSO ASG WP

... poprzednie wezly		WEZLY WYBRANE:	
1KA	5D	1KA	5D
1KA	1DZ	1KA	1DZ
1KA	2DZ	1KA	2DZ
1KA	3DZ	1KA	3DZ
1KA	4DZ		
2KA	5D		
2KA	1DZ		
2KA	2DZ		
2KA	3DZ		

nastepne wezly ...
KONIEC WYBIERANIA WEZLOW

Wybierz uzywajac klawiszy: Home, End, PgUp, PgDn.
Po wybraniu nacisnij Enter.

KSO ASG WP

maska 3

WEZLY		ZGLOSZENIA Szt	ZGLOSZENIA Czas	ZAKLOC. Szt	NIEZAKLOCONE CzR	NIEZAKLOCONE War	NIEZAKLOCONE ZajS	EFEKTYWNOSC %Zgl	EFEKTYWNOSC %Czasu	ZAKLOC. SrOpozn.
1KA SD		145	46241	51	48	29	17	35	51	18"
w tym:	UKF	70	16964	25	24	5	16	36	48	47"
	KF	44	17181	4	15	24	1	9	20	8"
	R/1	31	12095	22	9	0	0	71	99	166"
1KA 1DZ		116	29386	8	39	5	64	7	10	4"
w tym:	UKF	45	11846	5	15	0	25	11	13	6"
	KF	42	12314	3	17	5	17	7	11	11"
	R/1	29	5226	0	7	0	22	0	0	0"
1KA 2DZ		109	29495	21	35	7	46	19	28	14"
w tym:	UKF	57	12253	16	20	0	21	28	35	55"
	KF	36	11421	3	12	7	14	8	19	19"
	R/1	16	5822	2	3	0	11	13	31	113"
1KA 3DZ		93	27925	5	29	32	27	5	11	3"
w tym:	UKF	33	11676	4	12	1	16	12	25	26"
	KF	46	10819	0	15	31	0	0	0	0"
	R/1	14	5430	1	2	0	11	7	4	15"

Nacisnij klawisz!

KSO ASG WP

maska 4

	ZGLOSZENIA Szt	ZGLOSZENIA Czas	ZAKLOC. Szt	NIEZAKLOCONE CzR	NIEZAKLOCONE War	NIEZAKLOCONE ZajS	EFEKTYWNOSC %Zgl	EFEKTYWNOSC %Czasu	ZAKLOC. SrOpozn.
Razem:	463	133047	85	151	73	154	18	28	3"
w tym:UKF	205	52739	50	71	6	78	24	32	10"
KF	168	51735	10	59	67	32	6	13	3"
R/1	90	28574	25	21	0	44	28	49	21"

WYBRANA GRUPA WEZLOW:

1KA SD 1KA 1DZ 1KA 2DZ

1KA 3DZ

Nacisnij klawisz!