ETAPY ROZWIĄZYWANIA PROBLEMÓW

1. Sformułowanie zadania.
2. Określenie danych wejściowych.
3. Określenie celu, czyli wyniku.
4. Poszukiwanie metody rozwiązania, czyli algorytmu.
5. Przedstawienie algorytmu w postaci:
    - opisu słownego;
    - listy kroków;
    - schematu blokowego;
    - pseudokodu;
    - jednego z języków programowania;
6. Analiza poprawności rozwiązania.
7. Testowanie rozwiązania dla różnych danych - ocena efektywności przyjętej metody.
8. Modyfikacja i uzupełnianie.

SPECYFIKACJA ZADANIA

Opisanie zadania.

Określenie danych i szukanych oraz związku, jaki zachodzi między danymi a wynikami (szukanymi).

ALGORYTM

Ściśle określony sposób rozwiązania problemu (zadania).

Przedstawienie rozwiązania zadania w sposób uporządkowany, tj. z wyszczególnieniem kolejnych czynności.

SPOSOBY ZAPISYWANIA ALGORYTMU

1. ZAPIS ALGORYTMU W POSTACI LISTY (CIĄGU) KROKÓW polega na podaniu kolejnych wykonywanych operacji, składających się na całe rozwiązanie.

2. ZAPIS ALGORYTMU W POSTACI GRAFICZNEJ - SCHEMATY BLOKOWE.

Schemat blokowy to graficzne przedstawienie ciągu kroków algorytmu, często stosowane jako ideowy rysunek poprzedzający tworzenie programu.

W schemacie blokowym poszczególne operacje przedstawiane są za pomocą odpowiednio połączonych skrzynek (klocków, bloków).

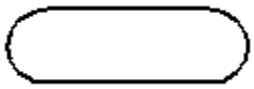
Sposób i kolejność działań programu określane są przez wzajemny układ i sposób łączenia bloków ze sobą. Każde działanie (krok) ma w schemacie blokowym swoje standardowe oznaczenie.

2.1. ZASADY BUDOWY SCHEMATU BLOKOWEGO

- każda operacja jest umieszczona w skrzynce;
- schemat ma tylko jedną skrzynkę „początek” („start”) i przynajmniej jedną skrzynkę „koniec” („end”);
- skrzynki są ze sobą połączone;
- ze skrzynki wychodzi jedno połączenie; wyjątek stanowią skrzynki: „koniec” (z której nie wychodzą już żądne połączenia) oraz „warunkowa” (z której wychodzą dwa połączenia opisane TAK i NIE - w zależności od tego, czy warunek jest spełniony czy nie, można wyjść jedną z dwóch dróg);

2.2. ELEMENTY (KLOCKI) SCHEMATU BLOKOWEGO

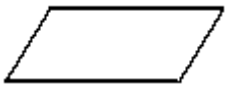
Poszczególne elementy schematu łączy się za pomocą strzałek. W większości przypadków blok ma jedną strzałkę wchodzącą i jedną wychodzącą, lecz są także wyjątki.



Ta figura oznacza początek lub koniec algorytmu. W każdym algorytmie musi się znaleźć dokładnie jedna taka figura z napisem „Start” oznaczająca początek algorytmu oraz dokładnie jedna figura z napisem „Stop” oznaczająca koniec algorytmu. Najczęściej popełnianym błędem w schematach blokowych jest umieszczanie kilku stanów końcowych, zależnych od sposobu zakończenia programu. Jest to niedopuszczalne, w programie mamy przecież dokładnie jedną instrukcję „end”. Blok symbolizujący początek algorytmu ma dokładnie jedną strzałkę wychodzącą a blok symbolizujący koniec ma co najmniej jedną strzałkę wchodzącą.

Jest to figura oznaczająca proces (operację, działanie). W jej obrębie umieszczamy wszelkie obliczenia lub podstawienia. Proces ma dokładnie jedną strzałkę wchodzącą i dokładnie jedną strzałkę wychodzącą.

Romb symbolizuje blok decyzyjny. Umieszcza się w nim jakiś warunek (np. "x>2"). Z dwóch wybranych wierzchołków rombu wyprowadzamy dwie możliwe drogi: gdy warunek jest spełniony (strzałkę wychodzącą z tego wierzchołka należy opatrzyć etykietą "Tak") oraz gdy warunek nie jest spełniony. Każdy romb ma dokładnie jedną strzałkę wchodzącą oraz dokładnie dwie strzałki wychodzące.



Równoległobok jest stosowany do odczytu lub zapisu danych. W jego obrębie należy umieścić stosowną instrukcję np. Read(x) lub Write(x) (cin>>x lub cout<<x), można też stosować opis słowny np. „Drukuj x na ekran”. Figura ta ma dokładnie jedną strzałkę wchodzącą i jedną wychodzącą.

Ta figura symbolizuje proces, który został już kiedyś zdefiniowany. Można ją porównać do procedury, którą definiuje się raz w programie, by następnie móc ją wielokrotnie wywoływać. Warunkiem użycia jest więc wcześniejsze zdefiniowanie procesu. Podobnie jak w przypadku zwykłego procesu i tu mamy jedno wejście i jedno wyjście.

Koło symbolizuje tzw. łącznik stronicowy. Może się zdarzyć, że chcemy „przeskoczyć" z jednego miejsca na kartce na inne (np. by nie krzyżować strzałek). Możemy w takim wypadku posłużyć się łącznikiem. Umieszczamy w jednym miejscu łącznik z określonym symbolem w środku (np. cyfrą, literą) i doprowadzamy do nie go strzałkę. Następnie w innym miejscu kartki umieszczamy drugi łącznik z takim samym symbolem w środku i wyprowadzamy z niego strzałkę. Łącznik jest często porównywany do teleportacji (z jednego miejsca na kartce do drugiego). Łączniki występują więc w parach, jeden ma tylko wejście a drugi wyjście.

Ten symbol to łącznik międzystronicowy. Działa analogicznie jak pierwszy, lecz nie w obrębie strony. Przydatne w złożonych algorytmach, które nie mieszczą się na jednej kartce. Uwaga: jeśli stosujemy oba typy łączników w schemacie, to najlepiej jest stosować liczby do identyfikowania jednych i litery do drugich. Dzięki temu nie dojdzie do pomyłki.

WARUNEK LOGICZNY

z sytuacją warunkową mamy do czynienia wówczas, gdy wynik lub dalsze działanie należy od spełnienia warunku. Na schemacie blokowym sytuacje warunkowe realizujemy przez skrzynkę warunkową.

WARUNEK to wyrażenie logiczne, którego wartość (TAK lub NIE, PRAWDA lub FAŁSZ, TRUE or FALSE, 0 lub 1) decyduje o wykonaniu lub nie uwarunkowanej nim instrukcji.

ITERACJA (działanie w pętli) to technika algorytmiczna polegająca na wykonaniu tej samej instrukcji dla n zmiennych (np. liczb, wartości logicznych etc.). Iteracja oszczędza czas programisty, który ten musiałby spędzić wpisując instrukcję n razy, zależnie od liczby zmiennych. Liczba powtórzeń w iteracji jest z góry określona lub zależy od spełnienia określonego warunku.

W realizacji algorytmów iteracyjnych ważne jest prawidłowe określenie sposobu zakończenia działań. Można to zrobić za pomocą licznika, który odlicza kolejne kroki iteracji (liczbę powtórzeń).

Zapętlenie algorytmu to błąd w projektowaniu kroków algorytmu polegający na pominięciu warunku (np. licznika), który kończy działanie w pętli (iterację). Wtedy iteracja ciągnie się w nieskończoność, gdyż program wykonujący działanie nie wie, kiedy ma ją przerwać (czyt. „komputer się zawiesił").

REKURENCJA (rekursja) to odwoływanie się do samej/samego siebie (wywoływanie samego siebie)

PROGRAM to ciąg instrukcji przetłumaczonych na język „wewnętrzny" komputera „zrozumiały" dla procesora wykonujący określony algorytm. Aby zatem napisać własny program, należy poznać nie tylko instrukcje programowania, ale przede wszystkim metody programowania.

Aby przedstawić algorytm w postaci programu, trzeba go napisać jako ciąg instrukcji języka programowania. Każda instrukcja, podobnie jak skrzynka w schemacie blokowym,

odpowiada określonej operacji, dlatego też kolejność występowania instrukcji w programie określa kolejność wykonywania operacji.

Własne programy piszemy posługując się językami programowania, takimi jak Pascal, język C/C++, Java, Basic… Do programowania służą programy - specjalne edytory wchodzące w skład środowiska programowania, zawierającego zwykle oprócz edytora kompilator i inne narzędzia wspomagające programowanie, np. Turbo Pascal, Turbo C++, Visual Basic, Delphi, C++Builder.

TRANSLACJA to tłumaczenie programu na język wewnętrzny komputera, wykonywane za pomocą wyspecjalizowanego programu, tzw. translatora. Wyróżniamy dwa typy translacji: kompilację i interpretację.

Po napisaniu ciągu instrukcji w wybranym języku programowania należy zapisać program w pliku na nośniku pamięci zewnętrznej, np. dysku twardym oraz wykonać jego kompilację, czyli uruchomić proces tłumaczący instrukcje na język zrozumiały dla procesora. Po poprawnym przeprowadzeniu kompilacji można program uruchomić.

W zależności od języka programowania i wersji programu służącego do pisania programów plik utworzony w takim programie może mieć różne rozszerzenia, np. programy pisane w Turbo Pascalu mają rozszerzenie pas, w C++ cpp.

KOMPILACJA - przetłumaczenie napisanego przez nas programu w całości, tak by mógł on być wykonany przez komputer przy każdorazowym uruchomieniu. Raz skompilowany program nie wymaga już powtórnej operacji tłumaczenia. Do wykonania kompilacji służą programy narzędziowe - kompilatory.

INTERPRETACJA - tłumaczenie programu tworzonego w jednym z języków programowania instrukcja po instrukcji, tak by każda wywołana instrukcja była wykonana przez komputer. Tłumaczenie następuje każdorazowo przy uruchomieniu programu.

KOMÓRKA PAMIĘCI: komputer, aby móc bez przeszkód poruszać się po swoim skomplikowanym wnętrzu, potrzebuje wielu danych o własnej „anatomii”. Wszystkie miejsca ważne dla jego działania (procesor, dysk, sloty kart rozszerzenia etc.) mają przypisany sobie

fragment pamięci RAM. Pamięć ta podzielona jest na logiczne komórki, z których każda posiada swój adres. Dzięki temu pisząc programy, możemy nakazać im przebywanie i korzystanie z określonych obszarów pamięci RAM.

PROGRAMOWANIE STRUKTURALNE to rozbicie działania programu na procedury/funkcje (podprogramy), z których każda odpowiada za rozwiązanie określonego problemu. Procedury/funkcje stanowią wtedy odrębne, samodzielnie działające całości, które możemy wykorzystać także i w innych pisanych programach.

PROCEDURA/FUNKCJA to wyodrębniona część programu, mająca jednoznaczną nazwę i ustalony sposób wymiany danych z innymi częściami programu.

PARAMETRY to zmienne, poprzez które procedura komunikuje się z innymi fragmentami programu. Parametry formalne to zmienne wpisane w procedurę/funkcję, które zastępowane są przez konkretne dane (parametry aktualne) w momencie wywołania programu.

PODSUMOWANIE I UZUPEŁNIENIE

- ALGORYTMIKA to jeden z obszarów tematycznych informatyki; zajmuje się budowaniem teoretycznych modeli informatycznych, czyli algorytmami.
- ALGORYTM podaje krok po kroku sposób rozwiązania problemu.
- Rozwiązując dowolny problem, postępujemy według podobnego schematu; określamy: dane wejściowe, sposób ich przetwarzania i wyniki, czyli cel i sposób rozwiązania.
- W schemacie blokowym, czyli graficznej prezentacji algorytmu, układ skrzynek i połączeń między nimi wyznacza kolejność i sposób wykonywania działań.
- W każdej skrzynce należy umieszczać jeden rodzaj operacji; skrzynki muszą być połączone.
- Dla czynności wielokrotnie powtarzanych stosuje się działania w pętli, czyli iterację, jedną z najczęściej spotykanych technik algorytmicznych.
- Stosując iterację należy określić sposób jej zakończenia, gdyż w przeciwnym razie program się zapętli. Zakończenie pętli może być uzależnione od zadanej liczby powtórzeń lub od spełnienia warunku logicznego.
- Program to logicznie uporządkowany ciąg instrukcji realizujący określone zadanie, czyli algorytm.
- Programowanie polega na przedstawieniu algorytmu w postaci instrukcji języka programowania, zapisanych w kolejności wyznaczonej przez ten algorytm.
- Instrukcja języka to zapis danej operacji w składni odpowiedniej dla danego języka programowania.
- Programy napisane w języku wysokiego poziomu, np. Turbo Pascal, C/C++, Visual Basic, nie są zrozumiałe dla komputera - są tłumaczone na język wewnętrzny komputera.
- Tłumaczenie programu z języka wysokiego poziomu na język wewnętrzny komputera nazywamy translacją.
- Pamięć komputera podzielona jest na mniejsze części (komórki). Każda z nich ma swój adres.
- Zmienna w programie ma określoną bieżącą wartość, która może ulegać zmianie w trakcie wykonywania zadania (po podstawieniu do niej innej wartości). Ze zmienną związane jest jej miejsce w pamięci komputera.
- Podprogramy (procedury/funkcje) rozwiązują wyodrębnione części zadania i mogą być wielokrotnie wywoływane przez dany program (lub inny).

- Główne metody sortowania to: przez wybór, bąbelkowe, szybkie.
- Z rekurencją spotykamy się w definicjach, w których definiowane pojęcie odwołuje się do samego siebie.
- Algorytmy iteracyjne prawie zawsze mogą być zapisane w postaci rekurencji i odwrotnie.

CECHY ALGORYTMÓW

kompletność

musi uwzględniać wszystkie możliwe przypadki, które mogą pojawić się podczas jego realizacji

poprawność

dla każdego zestawu danych, po wykonaniu skończonej liczby czynności, prowadzi do poprawnych wyników
(algorytm daje dobre wyniki)

jednoznaczność

w każdym przypadku jego zastosowania, dla tych samych danych uzyskamy ten sam wynik,
(daje takie same wyniki przy takich samych danych)

skończoność

musi zapewnić osiągnięciedo rozwiązania w skończonej liczbiekroków
(wykonuje się w skończonej ilości kroków, w skończonym czasie)

uniwersalność

służy do rozwiązywania pewnej grupy zadań, a nie tylko jednego zadania (np. algorytm jest przepisem na rozwiązanie równania postaci ax + b = 0 dla dowolnych współczynników a i b, a nie - jednego konkretnego równania, np. 2x + 3 = 0.

sprawność (efektywność, złożoność: obliczeniowa, czasowa i pamięciowa)
prowadzenie do rozwiązania najkrótszą drogą