# Wskaźniki

## Wskaźniki

Oprócz zmiennych różnego typu, które do tej pory poznaliśmy w języku programowania C++ wyróżniamy jeszcze tzw. wskaźniki. Wskaźnik to zmienna przechowująca adres innej zmiennej. Inaczej, jest to zmienna wskazująca na adres innej zmiennej danego typu.

Poniżej podajemy definicję wskaźnika mogącego pokazywać na obiekty typu int.

int \*w;

Nasz wskaźnik nazywa się w (bez gwiazdki), gwiazdka informuje nas tylko o tym, że w jest właśnie wskaźnikiem. Wracając do definicji powiemy, że we wskaźniku w możemy przechowywać adres jakiegoś obiektu typu int. Zatem treścią wskaźnika jest informacja o tym **gdzie** wskazywany obiekt się znajduje a nie **co** w nim się znajduje.

Podsumowując wszystkie zmienne, jakie deklarujemy są niczym innym jak tylko komórkami pamięci operacyjnej RAM (może być inaczej, ale nie jest to teraz istotne). Zatem odwołując się do zmiennej poprzez jej nazwę odwołujemy się do przydzielonej jej (zmiennej) pamięci. Wartością wskaźnika jest natomiast adres pamięci RAM, gdzie znajduje się taka zmienna.

Wskaźniki mogą być do obiektów różnych typów: np. char, float, jak również do obiektów, które sami zdefiniujemy. Zapamiętajmy jednak, że wskaźnik służący do pokazywania na obiekty jednego typu nie nadaje się do pokazywania na obiekty innego typu.

Najczęściej wskaźniki stosuje się w sytuacjach, gdy chodzi o

- ulepszenie pracy z tablicami,

- funkcje mogące zmieniać wartość przesyłanych do nich argumentów,

- dostęp do specjalnych komórek pamięci,

- rezerwację obszarów pamięci,

- pracę ze strukturami danych.

### Praca ze wskaźnikami

Ze wskaźnikami możemy pracować podobnie jak z innymi znanymi nam zmiennymi. Innymi słowy możemy odczytać jego zawartość czy też zapisać coś do niego. Poniższy przykład ilustruje podstawowe operacje z użyciem wskaźnika, czyli deklaracja wskaźnika, ustawienie wskaźnika na dany obiekt & (ampersand), nadanie wskaźnikowi wartości oraz jak wypisanie informacji zawartej we wskaźniku.

1. #include<iostream>
2. #include<stdlib.h>
3. using namespace std;
4. int main()
5. {
6. int x=1; //deklaracja zmiennej typu int
7. int \*wskaznik; //deklaracja wskaźnika na typ int
8. wskaznik = & x; //ustawienie wskaźnika na obiekt x
9. \*wskaznik = 99; //zapis równoważny z "x=99;"
10. cout<<"W obiekcie pokazywanym przez wskaźnik jest wartosc "<<(\*wskaznik);
11. cout<<endl;
12. system("Pause");
13. return 0;
14. }

**Listing 8.1. Nadawanie wartości wskaźnikowi**

Po wykonaniu powyższego programu na ekranie zobaczymy następujący efekt.

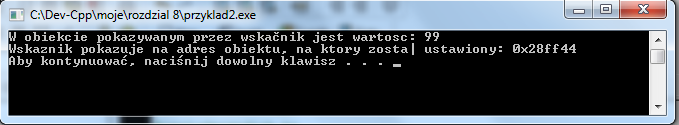
**Rys 8.1 Wynik działania programu z listingu 8.1**

Jak widzimy na przykładzie gwiazdka kieruje nas do obiektu pokazywanego przez wskaźnik. Ponieważ wskaźnik został ustawiony na obiekcie x, zatem od tej pory \*wskaznik oznacza to samo, co x. Zatem do danego obiektu możemy odnosić się na dwa sposoby: albo przez jego nazwę albo przez zdefiniowanie wskaźnika, który na ten obiekt pokaże.

Jednak **uwaga !** sama **nazwa** wskaznik (bez gwiazdki), jak już wcześniej było podane, oznacza adres gdzie zmienna x się znajduje a nie, co się w niej znajduje. Zobaczmy, co zostanie pokazane na ekranie, jeśli w przykładzie 1 dodamy instrukcję wypisania zawartości wskaźnika wskaznik (bez gwiazdki).

1. #include<iostream>
2. #include<stdlib.h>
3. using namespace std;
4. int main()
5. {
6. int x=1;
7. int \*wskaznik;
8. wskaznik = & x;
9. \*wskaznik = 99;
10. cout<< "W obiekcie pokazywanym przez wskaźnik jest wartosc: "<<(\*wskaznik)<<endl;;
11. cout<< "Wskaznik pokazuje na adres obiektu, na ktory został ustawiony: " <<(wskaznik);
12. cout<<endl;
13. system("Pause");
14. return 0;
15. }

**Listing 8.2 Podstawowe operacje z użyciem wskaźników.**



**Rys 8.2 Wynik działania programu z listingu 8.2**

Aby uniknąć błędów zwracajmy uwagę na operacje, które chcemy wykonywać za pomocą wskaźników, czy chcemy odnosić się do obiektu, na który pokazuje wskaźnik czy też chcemy korzystać z adresu tego obiektu.

Zgodnie z tym, co zostało napisane wcześniej wskaźniki mogą służyć do pokazywania na obiekty różnych typów, również tych, które sobie sami zdefiniujemy. Zobaczmy na przykładzie.

**Przykład 3.**  Napisz program definiujący zmienną typu TData oraz wskaźnik do zmiennej typu TData. Program powinien wczytać z klawiatury wartości pól zmiennej.

1. #include<iostream>
2. #include<stdlib.h>
3. using namespace std;
4. struct TData{
5. int d,m,r;
6. };
7. int main(int argc, char\* argv[])
8. {
9. TData data;
10. TData \*wsk;
11. wsk = &data; //ustawienie wskaźnika na obiekcie data typu TData
12. cout << "\ndzien=";
13. cin >> (\*wsk).d;
14. cout << "\nmiesiac=";
15. cin >> (\*wsk).m;
16. cout << "\nrok=";
17. cin >> (\*wsk).r;
18. cout << "\n\n======\n\n";
19. cout << (\*wsk).d << "-" << (\*wsk).m << "-" <<
20. (\*wsk).r << endl;
21. system("Pause");
22. return 0;
23. }

**Listing 8.3. Zastosowanie wskaźników do struktur.**

Analizując kod powyższego programu należy zwrócić uwagę na sposób odwoływania się do poszczególnych pól obiektu typu Tdata w momencie wypisywania danych na monitor. Nie jest to jednak jedyny sposób uzyskiwania wartości z pól struktury, do osiągnięcia tego celu moglibyśmy użyć również następującego zapisu:

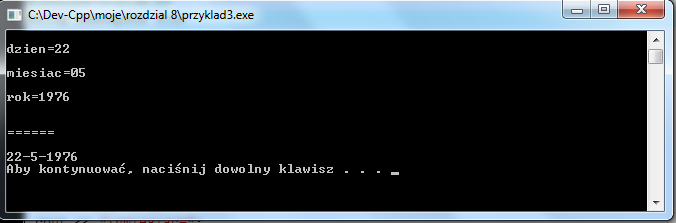
cout << wsk->d << "-" << wsk->m << "-" << wsk->r << endl;

Użycie nawiasu w powyższym listingu \*(wsk).d jest konieczne, ponieważ operator '.' (kropka) ma wyższy priorytet niż operator '\*'. Jeśli zatem wyrażenia '\*wsk' nie otoczylibyśmy nawiasami to byłoby to równoznaczne zapisowi:

\*(wsk.d) = 5; //to samo co: \*wsk.d=5;//Tutaj nawiasy zostały dodane aby pokazać jak zinterpretowałby to kompilator.

Kompilator wyrzuciłby oczywiście błąd, ponieważ najpierw chciałby on wyciągnąć pole 'd' od obiektu 'wsk', który nie jest obiektem typu TData ale wskaźnikiem na taki obiekt.  
Operator '->' pozwala nam tak samo na wywoływanie poszczególnych pól w strukturze

int x = wsk->d; //równoważne (\*wsk).d;

****

**Rys 8.3 Wynik działania programu z listingu 8.3**

### Dynamiczne tworzenie obiektów. Operatory new i delate

Pracując ze wskaźnikami nie zawsze mamy do dyspozycji nazwę obiektu, na który chcemy pokazywać wskaźnikiem. Tutaj z pomocą przychodzą nam operatory new i delate. Operator new zajmuje się tworzeniem obiektu( dynamicznie) natomiast operator delate służy do usuwania obiektów utworzonych w sposób dynamiczny.

Zauważmy, że jeśli mamy zdefiniowany wskaźnik np.

char \*w;

to następna instrukcja

w= new char

spowoduje utworzenie nowego obiektu typu char. Nie posiada on nazwy, ale jego adres przekazany został wskaźnikowi w. Od tego momentu używając jedynie wskaźnika możemy do tak utworzonego obiektu wprowadzać dane jak również możemy te dane z niego usuwać. Możemy również usunąć cały obiekt pokazywany przez wskaźnik w : służy do tego instrukcja

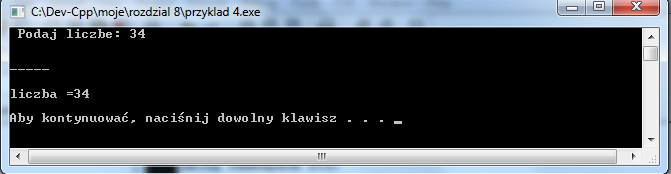
delate w;

Użycie opisanych powyżej operatorów zostało pokazane w następującym przykładzie.

**Przykład 4**. Napisz program, który dynamicznie zaalokuje zmienną typu int. Program powinien wczytać z klawiatury wartość i podstawić ją do zaalokowanej zmiennej. Następnie program powinien wydrukować na ekranie wartość zmiennej, po czym zwolnić pamięć i zakończyć działanie

1. #include<iostream>
2. #include<stdlib.h>
3. using namespace std;
4. int main()
5. {
6. int \*wsk;
7. wsk = new int;
8. cout << " Podaj liczbe: ";
9. cin >> \*wsk;
10. cout << "\n\n-----\n\n";
11. cout << "liczba =";
12. cout << \*wsk;
13. cout << endl << endl;
14. delete wsk;
15. system("Pause");
16. return 0;
17. }

**Listing 8.4. Użycie operatora new i delate**



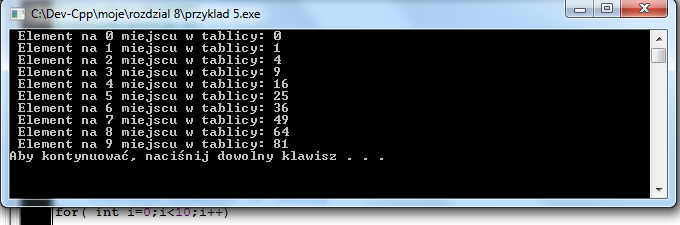
**Rys 8.4. Wynik działania programu z listingu 8.4**

### Wskaźniki a tablice

Zacznijmy od przeanalizowania następującego przykładu. Mamy zdefiniowaną tablicę np. liczb naturalnych postaci: int tab[10]; Podstawowe operacje na tej tablicy możemy wykonywać właśnie za pomocą wskaźników.

1. #include<iostream>
2. #include<stdlib.h>
3. using namespace std;
4. int main()
5. {
6. int \*wtab;
7. int tab[10];
8. wtab=&tab[0];//ustawienie wskaźnika na początku tablicy
9. // dodawanie elementów do tablicy
10. for( int i=0;i<10;i++)
11. {
12. \*(wtab)=i\*i;
13. wtab++;
14. }
15. wtab=tab;//ustawienie wskaźnika na początku tablicy
16. // wypisanie elementów tablicy
17. for(int i=0;i<10;i++)
18. {
19. cout<<" Element na "<< i<<" miejscu w tablicy: "<<(\*wtab)<<endl;
20. wtab++;
21. }
22. system("Pause");
23. return 0;
24. }

**Listing 8.5. Operacje na tablicy z wykorzystaniem wskaźników**



**Rys 8.5. Wynik działania programu z listingu 8.5**

Analizując powyższy program zauważmy, że wskaźniki możemy przesuwać. Jeśli ustawimy wskaźnik na początku tablicy to wykonując operacje dodawania lub odejmowania możemy poruszać się po elementach tablicy.

**Uwaga!** Zapis wtab=&tab[0]; jest równoważny wtab=tab i oznacza ustawienie wskaźnika na początku tablicy.

Wniosek :

**Nazwa tablicy jest jednocześnie adresem jej zerowego elementu.**

Powyższy wniosek jest szczególnie przydatny przy przekazywaniu tablic do funkcji., ale o tym w późniejszych rozdziałach.

## Tablice dynamiczne

### Dynamiczne tablice jednowymiarowe

Za pomocą operatora new możemy tworzyć nie tylko pojedyncze obiekty, jak to zostało pokazane w podrozdziale 8.1.2, ale również tablice.

int \* tablica;

tablica = new int[rozmiar];

gdzie rozmiar jest wyrażeniem typu int.

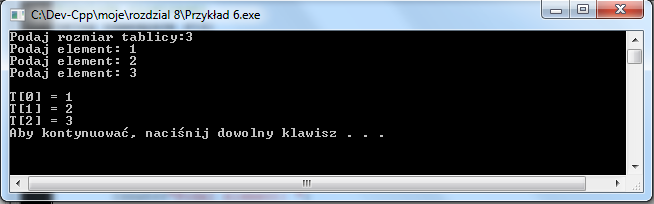
lub

int \*tablica= new int [rozmiar];

Zauważmy, że rozmiar tablicy nie musi być stałą znaną w momencie kompilacji, jak to miało miejsce w przypadku deklaracji tablic statycznych. Operator new daje nam możliwość swobodnego definiowania tablicy w trakcie wykonywania programu. Poniższy program pokazuje nam dynamiczną alokację tablicy jednowymiarowej.

1. #include <iostream>
2. using namespace std;
3. main()
4. {
5. int \*T; // tworzymy wskaźnik
6. int i,n;
7. cout<<"Podaj rozmiar tablicy:";
8. cin >> n; // odczytujemy ilość komórek
9. T = new int[n]; // tworzymy tablicę dynamiczną o n komórkach
10. for(i = 0; i < n; i++)
11. {cout<<"Podaj element: ";
12. cin >> T[i]; // wczytujemy kolejne komórki
13. }
14. for(i = 0; i < n; i++)// wypisujemy odczytaną tablicę
15. cout << endl << "T[" << i << "] = " << T[i];
16. cout << endl;
17. system("PAUSE");
18. }

**Listing 8.6. Dynamiczna alokacja tablicy jednowymiarowej**

****

**Rys 8.6. Wynik działania programu z listingu 8.6**

Analizując powyższy program zauważmy, że każdą zadeklarowaną w ten sposób tablicę należy „zlikwidować” tzn. zwolnić zajętą przez nią pamięć:

delete [] tablica;

**Uwaga!** W przypadku tworzenia obiektów za pomocą operatora new istnieje niebezpieczeństwo utraty „kontroli” nad danym obiektem, gdyż nie ma on nazwy i możemy się do niego donosić tylko za pomocą wskaźników. Utrata kontaktu z danym obiektem uniemożliwia skasowanie go a wówczas jeśli tworzymy dużo obiektów ( np. w pętli) , które nie zostaną skasowane to rodzi się kolejne niebezpieczeństwo związane z wyczerpaniem przyznanego nam obszaru pamięci a co za tym idzie nie będziemy mogli utworzyć już żadnych obiektów w naszym programie. Zatem jeśli w programie zamierzamy kreować dużo obiektów dynamicznych korzystając z zapasu pamięci to musimy pamiętać, że pamięć się w końcu wyczerpie. Trzeba się z tym liczyć i sprawdzać czy operacja się powiodła

float \*wsk;

Wsk=new float[8192];

if(!wsk) //czyli if(wsk==NULL)

{

cout<<”Pamięć się wyczerpała”;

}

### Dodawanie/usuwanie elementów w tablicy dynamicznej

Nowy element dodajemy – usuwanie odbywa się według podobnego algorytmu – w sposób następujący:

1. Tworzymy kopię tablicy o jeden element większą,
2. Przepisujemy elementy z tablicy źródłowej,
3. Dodajemy nowy element,
4. Usuwamy starą tablicę,
5. Kopiujemy kopię do starej tablicy,
6. Usuwamy kopię.

Cały program wygląda następująco:

#include <iostream>

using namespace std;

int main()

{

int \*T;

int i,n;

cout<<"Podaj rozmiar tablicy:";

cin >> n;

T = new int[n];

for(i = 0; i < n; i++)

{

cout<<"Podaj element: ";

cin >> T[i];

}

for(i = 0; i < n; i++)

cout << endl << "T[" << i << "] = " << T[i];

cout << endl;

//dodanie/usuwanie elemntów w tablicy dynamicznej

//Algorytm: tworzymy kopię o 1 większą

int \*Tkopia;

Tkopia=new int [n+1];

//przepisujemy elementy

for (int i=0;i<n;i++)

Tkopia[i]=T[i];

//dodajemy nowy element

Tkopia[n]=100;

//usuwamy starą tablicę

delete [] T;

//kopiujemy kopię do starej tablicy

T=Tkopia;

//usuwamy kopię

delete [] Tkopia;

for(int i=0;i<n+1;i++)

cout << endl << "T[" << i << "] = " << T[i];

}

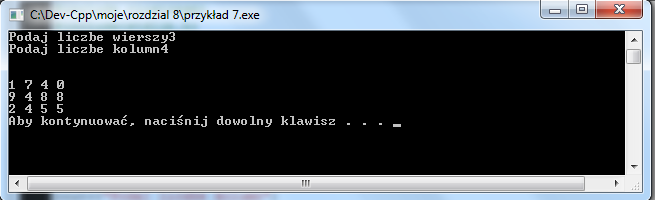
Uwaga: elementy do tablicy można dodawać za pomocą funkcji realloc, te zagadnienia zostaną omówione na wykładzie.

### Dynamiczne tablice wielowymiarowe

Analogicznie jak w przypadku tablic statycznych tak również w sposób dynamiczny możemy tworzyć tablice wielowymiarowe np. dwuwymiarowe. Przeanalizujmy to na następującym przykładzie.

1. #include <iostream>
2. #include<stdlib.h>
3. using namespace std;
4. main()
5. {
6. int \*\*tablica;
7. int l\_wierszy,l\_kolumn;
8. cout<<"Podaj liczbe wierszy";
9. cin>>l\_wierszy;
10. cout<<"Podaj liczbe kolumn";
11. cin>>l\_kolumn;
12. cout<<endl;
13. tablica=new int\*[ l\_wierszy];
14. for(int i=0;i< l\_wierszy;i++)
15. tablica[i]=new int[l\_kolumn];
16. //Powstała tablica i możemy ją teraz wypełnić danymi
17. for(int i=0;i< l\_wierszy; i++)
18. for(int j=0;j< l\_kolumn; j++)
19. {
20. tablica[i][j]= rand()%10;
21. }
22. cout<<endl;
23. for(int i=0;i< l\_wierszy; i++)
24. { for(int j=0;j< l\_kolumn; j++)
25. {
26. cout<<tablica[i][j]<<" ";
27. }
28. cout<<endl;
29. }
30. for(int i=0;i< l\_wierszy;i++)// zwolnienie pamięci
31. delete []tablica [i];
32. delete []tablica;
33. system("PAUSE");
34. }

**Listing 8.7. Dynamiczna alokacja tablicy dwuwymiarowej**



**Rys 8.7. Wynik działania programu z listingu 8.7**

Analizując powyższy przykład zwróćmy uwagę na sposób dynamicznej alokacji tablicy dwuwymiarowej.

Na początek dynamicznie tworzymy jednowymiarową tablicę wskaźników

tablica=new int\*[ l\_wierszy];

a następnie w pętli, w każdym elemencie powstałej tablicy jednowymiarowej tworzymy dynamicznie kolejne tablice jednowymiarowe

for(int i=0;i< l\_wierszy;i++)

tablica[i]=new int[l\_kolumn];

Analogicznie możemy stworzyć dynamiczną tablicę wielowymiarową.

## Zadania na „ Laboratorium programowania ”

1. Napisz program, w który utworzy dynamiczną tablicę dwuwymiarową liczb całkowitych o wymiarach podanych przez użytkownika. Następnie wypełni ją losowymi liczbami z przedziału [100, 300]. Wydrukuj na ekranie wszystkie liczby, których suma cyfr jest podzielna przez 3.
2. Napisz program, który utworzy dynamiczną dwuwymiarową tablicę liczb całkowitych o wymiarach podanych przez użytkownika. Następnie wypełni ją losowymi liczbami z przedziału [0, 200]. Znajdź indeks największego elementu w tablicy. Wypisz ten element i jego indeks.
3. Napisz program, który wczyta n –napisów i zapisze w dynamicznej tablicy dwuwymiarowej napisy wg. następującej reguły:

* pierwszy wiersz - napisy dłuższe niż 10 znaków,
* drugi wiersz – napisy rozpoczynające się wielką literą,
* trzeci wiersz – napisy, które zawierają cyfrę.

1. Napisz program, który wykorzystując Stos sprawdzi czy podany przez użytkownika wyraz jest palindromem.
2. Napisz program wczytujący podane przez użytkownika wyrażenie arytmetyczne zbudowanie z wykorzystaniem operatorów + - \* / () i zamieniający je na wyrażenia w notacji postfiksowej z wykorzystaniem stosu.
3. Napisz program wykorzystujący wzór Hornera do obliczenia wartości wielomianu stopnia n. Użyj stosu.
4. Napisz program, który na podstawie implementacji stosu utworzy listę jednokierunkową dla, której będą realizowane następujące zadania:

* Dodawanie na początek listy,
* Wstawianie elementu za znalezionym elementem w liście,
* Usuwanie elementu.

1. Napisz program, który na podstawie implementacji stosu będzie dodawał liczby z zachowaniem porządku.

**Wejscie**

4 5 2 7 6 9 3

**Wyjscie**

2 3 4 5 6 7 9

1. Napisz program, który losuje n liczb całkowitych i zapisuje je na stosie. Na podstawie wylosowanych danych utwórz dwa stosy, jeden z liczbami parzystymi, drugi z nieparzystymi. Wypisz listę i obydwa stosy z początkowymi danymi i wynikowymi.

## Zadania do samodzielnego rozwiązania

1. Napisz program, w którym stworzysz dynamiczną tablicę jednowymiarową, o wymiarze podanym przez użytkownika. Następnie wypełnij ją znakami. Wydrukuj na ekranie wszystkie samogłoski.
2. Zadeklaruj jednowymiarową dynamiczną tablicę liczb rzeczywistych, do przechowywania kursu $ w podanej przez użytkownika liczbie dni.
   1. Wprowadź dane do tablicy
   2. Wypisz dane z tablicy
   3. Oblicz średni kurs
3. Zadeklaruj dwuwymiarową dynamiczną tablicę liczb rzeczywistych o wymiarach m x n. Napisz program, który wypełni tablicę liczbami pseudolosowymi z przedziału -5, 5 i wypisze wartość i indeksy elementu tablicy najmniej różniącego się od średniej liczonej dla całej tablicy.
4. Używając tablicy dynamicznej napisz program dodający element na jej końcu lub usuwający. Wyboru dokonuje użytkownik w czasie działania programu. Wyświetl tablicę przed i po operacji.
5. Używając tablicy dynamicznej napisz program dodający element na jej początku lub usuwający pierwszy element. Wyboru dokonuje użytkownik w czasie działania programu. Wyświetl tablicę przed i po operacji.
6. Używając tablicy dynamicznej napisz program dodający element w miejscu określonym przez użytkownika programu. Wyboru dokonuje użytkownik w czasie działania programu. Wyświetl tablicę przed i po operacji. Sprawdź poprawność położenia miejsca wstawiania.
7. Używając tablicy dynamicznej napisz program usuwający element z miejsca określonego przez użytkownika programu. Wyboru dokonuje użytkownik w czasie działania programu. Wyświetl tablicę przed i po operacji. Sprawdź poprawność położenia miejsca usuwania.
8. Napisz program, który utworzy dynamiczną jednowymiarową tablicę o maksymalnym wymiarze 25. Do tablicy wczytaj kolejno n-liczb podanych przez użytkownika. Po podaniu wszystkich liczb do tablicy liczby ujemne należy zastąpić zerami, a następnie wydrukować wszystkie liczby na ekranie. Pamięć dla tablicy ma być przydzielona dynamicznie.
9. Napisz program, w którym:

* Stworzysz tablicę (macierz) o rozmiarze podanym przez użytkownika, z zakresu od [2 , 10], a następnie wypełnisz ją losowymi liczbami całkowitymi z przedziału od [-10, 10].
* Dla każdej kolumny wyznaczysz wartość minimalną i przepiszesz je do tablicy jednowymiarowej w taki sposób, że liczba minimalna z j-tej kolumny znajdzie się w j-tej kolumnie tablicy jednowymiarowej.
* Wyznaczysz największą wartość na obu przekątnych macierzy (po jednej wartości dla każdej z dwóch przekątnych) i ustawisz na nie dwa wskaźniki i za ich pomocą wypiszesz te wartości na ekran.

1. Napisz program, który utworzy tablicę dwuwymiarową liczb całkowitych o rozmiarach wczytanych z klawiatury. Tabelę wypełnij losowymi liczbami z przedziału [1900,1999]. Następnie ustaw wskaźnik na liczbę maksymalną z całej tabeli i za jego pomocą wypisze jej wartość na ekran.
2. Zadeklaruj dynamiczną trójwymiarową tablicę elementów typu całkowitego wypełnij ją danymi a następnie podziel ją na K (podane przez użytkownika) tablic („plastrów”) względem jednego z wymiarów (również podanego przez użytkownika). Uwzględnij brak możliwości równego podziału tzn. np. tab[10][12][15] nie da się względem ostatniego wymiaru na 2 tablice.
3. Dany jest ciąg zakończony zerem. Napisz program, który zapisze je w tablicy dynamicznej. Tablica dynamiczna ma mieć ściśle dopasowaną liczbę elementów.
4. Dana jest lista pracowników o strukturze: nazwisko, stanowisko. Napisz program, który utworzy tablicę dynamiczną zawierająca tylko stanowisko, stanowiska w tablicy nie mogą się powtarzać.
5. Dana jest lista pracowników o strukturze: nazwisko, miejscowość, dochód zapisana w tablicy dynamicznej. Napisz program, który wypisze jakie średnie dochody osiągają mieszkańcy miejscowości występujących na liście.
6. Dana jest lista pracowników o strukturze: nazwisko, imię, miejscowość zapisana w tablicy dynamicznej. Napisz program, który utworzy dwa stosy, jeden zawiera dane kobiet, drugi mężczyzn. Wypisz dane w tablicy dynamicznej i zawartość stosów. Przyjmij, że imię kobiety kończy na literę 'a'.
7. Dana jest tablica dynamiczna dwuwymiarowa. Składa się ona z liczby wierszy oraz z liczby kolumn będących liczbami losowymi z przedziału <5,10>. Napisz program, który przepisze dane z tablicy dynamicznej na stos, który ma tyle elementów ile ma wierszy tablica, każdy taki stos tworzy nowy stos, który zawiera tyle elementów ile mają poszczególne kolumny (otrzymana struktura to stos, którego elementy są stosami).
8. Napisz program, który dla danych wyrazów zapisanych na stosie sprawdzi, czy wczytany napis znajduje się na stosie. Wypisz TAK, gdy jest i NIE, gdy tego elementu nie ma.

## Zadania z „ Podstaw programowania”

1. Zadeklaruj jednowymiarową dynamiczną tablicę liczb rzeczywistych, do przechowywania kursu zł w podanej przez użytkownika liczbie dni.
   1. Wprowadź dane do tablicy
   2. Wypisz dane z tablicy
   3. Oblicz średni kurs dla dolara
2. Napisz program, w którym utworzy dynamiczną tablicę jednowymiarową, o wymiarze podanym przez użytkownika. Następnie wypełnij ją znakami. Wydrukuj na ekranie wszystkie samogłoski.
3. Zadeklaruj dynamiczną jednowymiarową tablicę elementów typu rzeczywistego wypełnij ją danymi a następnie
   1. posortuj tablicę rosnąco i wypisz ją,
   2. posortuj tablicę malejąca i wypisz ją,
   3. wypisz medianę.

UWAGA: nie używamy funkcji sort.

1. Zadeklaruj dynamiczną dwuwymiarową tablicę elementów typu całkowitego wypełnij ją danymi (kolumnowo) a następnie
   1. zmień miejscami dwie kolumny wybrane przez użytkownika
   2. zamień miejscami dwa wiersze wybrane przez użytkownika
2. Współczynniki wielomianu zapisane są w postaci stosu, gdzie na wierzchołku znajduje się liczba an (przy n-tej potędze). Napisz program obliczania wartości tego wielomianu dla dowolnego x=x0. Skorzystaj ze schematu Hornera.
3. Napisz program, w którym stworzysz dynamiczną dwuwymiarową tablicę liczb rzeczywistych o wymiarach podanych przez użytkownika. Następnie wypełnij ją losowymi liczbami z przedziału [0,50]. Stwórz tablicę jednowymiarową o odpowiedniej ilości wierszy i przepisz do niej te liczby, które są większe od 25. W przypadku braku takich liczb powinien zostać wyświetlony odpowiedni komunikat.
4. Zadeklaruj dynamiczną, dwywymiarową tablicę znaków o wymiarach mxn. Napisz program, który wypełni tablicę losowo literami alfabetu łacińskiego, wypisze ją i wypisze te litery, które występują we wszystkich wierszach macierzy.
5. Dana jest lista towarów o strukturze: nazwa, ilość, cena jednostkowa, dostawca. Zapisz tą listę w postaci stosu. Napisz program, który na podstawie tego stosu utworzy dwie tablice dynamiczne, jedna zawiera dane o wartości towaru większe od średniej z wartości, druga pozostałe dane. Wypisz stos z danymi początkowymi oraz tablice wynikowe.