Dev-C++

Część 5
tablice, sortowanie, ciąg Fibonacciego

Ćwiczenia wykonane pisemnie (notatki w zeszycie)

Zapisz numer pytania (przed pytaniem np. Pytanie 1) poniżej zapisz treść pytania. Treść pytania podkreśl na **zielono** a pod treścią pytania zapisz odpowiedź.

Pytanie 1
Podaj definicję tablicy jednowymiarowej i dwuwymiarowej;
Pytanie 2
Jaką rolę spełnia deklaracja tablic;

TABLICE w C++.

Tablica jest to struktura złożona z elementów tego samego typu.
Elementy tablicy są skazywane przez **indeks** lub **zespół indeksów**. Określają one miejsce gdzie dany element znajduje się w tablicy.

---

tablica jednowymiarowa

1	**3**	**4**	**10**
A[0]	A[1]	A[2]	A[3]

w komórce jest umieszczona A[0]=1 A[1]=3 A[2]=4 A[3]=10

tablica dwuwymiarowa

A[0][0]	A[0][1]	A[0][2]	A[0][3]	A[0][4]	A[0][5]
3	**1**	**2**	**10**	**6**	**5**
1	**3**	**4**	**20**	**7**	**9**
A[1][0] Pierwsza kolumna	A[1][1]	A[1][2]	A[1][3]	A[1][4]	A[1][5] szósta kolumna

czyli w komórce jest np.
A[0][0]=3 A[1][4]=7 A[0][2]=2 A[1][5]=9
miejsce w tablicy określone jest przez podanie nazwy tablicy i w nawiasach kwadratowych podajemy wiersz potem po przecinku kolumnę.

Deklaracja tablic:
Tablice wymagają przed użycie deklaracji komputer musi wiedzieć ile miejsca zarezerwować w pamięci i w jaki sposób rozmieścić kolejne elementy tablicy.
Przykład

int TAB[15];

wytłumaczenie:
należy zarezerwować 15 kolejnych komórek pamięci dla 15 liczb całkowitych typu int. Jednowymiarowa tablica (wektor) będzie się nazywać "TAB", a jej kolejne elementy zostaną ponumerowane przy pomocy indeksu.
Należy zwróć uwagę, że w C++ zaczynamy liczyć numery komórek od zera a nie od jedynki.
Tablicz TAB wygląda następująco
TAB[0], TAB[1], TAB[2], TAB[3], ... TAB[14].

Przykład
float TAB_DWU[10][5];
Tablica TAB_DWU jest tablicą dwuwymiarową i składa się z 50 elementów typu rzeczywistego (10 wierszy i 5 kolumn)

Przykład
Nadawanie wartości początkowych elementom tablicy. Wartości takie należy podawać w nawiasach klamrowych.

const int b[4]={1,2,33,444};
Wytłumaczenie:
Elementom jednowymiarowej tablicy (wektora) b przypisano
wartości: b[0]=1; b[1]=2; b[2]=33; b[3]=444;

Przykład
Nadawanie wartości początkowych elementom tablicy dwuwymiarowej.

int TAB[2][3]={{1, 2, 3},{2, 4, 6}};

Przykład 23
Nadawanie wartości początkowych elementom tablicy znakowej.

char hej[5]={'A', 'h', 'o', 'j'};

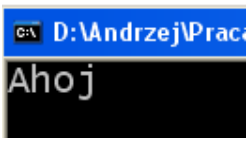
```
#include <windows.h>
#include <iostream>
#include <conio.h>
#include <cstdlib>

using namespace std;

int main()
{
   char hej[5]={'A', 'h', 'o', 'j'};
   cout << hej;                    //program wyświetli nam zawartość całej tablicy hej
   getch();
}
```

char hej[5]="Ahoj"; Jako piątym elementem tablica zostanie uzupełniona znakiem terminalnym - \0 zabezpiecza on miejsce na wstawianie pozostałych znaków;

Powyższą sytuację demonstruje również przykład

Przykład 24

```
#include <windows.h>
#include <iostream>
#include <conio.h>
#include <cstdlib>

using namespace std;

int main()
{
   char str[20];
   str[0] = 'A';
   str[1] = 'h';
   str[2] = 'o';
   str[3] = 'j';
   str[4] = '\0';

   cout << str;

   getch();
}
```

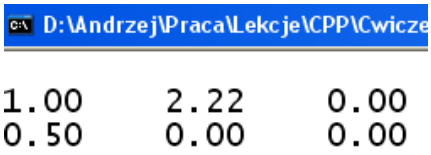


Przykład 25
Tablica uzupełniona zerami przez domniemanie.

float T[2][3]={{1, 2.22}, {.5}};
kompilator uzupełni zerami wartości pozostałych komórek.

```
#include <windows.h>
#include <iostream>
#include <conio.h>
#include <cstdlib>

using namespace std;

int main()
{
   float T[2][3] = {{1,2.22}, {.5}};
```

```
   cout << endl;
   for( int  i = 0; i < 2; i++ )
   {
      for( int j = 0; j < 3; j++ )
      {
         printf("%.2f\t", T[i][j]);
      }
      cout << endl;
   }
      getch();
}
```



czyli

T[0,0]	T[0,1]	T[0,2]
1.00	2.22	0.00
0.50	0.00	0.00
T[1,0]	T[1,1]	T[1,2]

Przykład
char D[ ]="Jakis napis"
int A[ ][2]={{1,2}, {3,4}, {5,6}}

Jeśli nawias kwadratowy zawierający wymiar pozostawimy pusty, to kompilator obliczy jego domniemaną zawartość w oparciu o podaną zawartość tablicy.

Przykład 26
Wykonaj:
1)Zapisz w zeszycie temat zadnia.
2)uruchom przykład
3)Zapisz w zeszycie siedem linijek wypisywania tablicy po jej wczytywaniu, czyli działanie drugiej pętli.

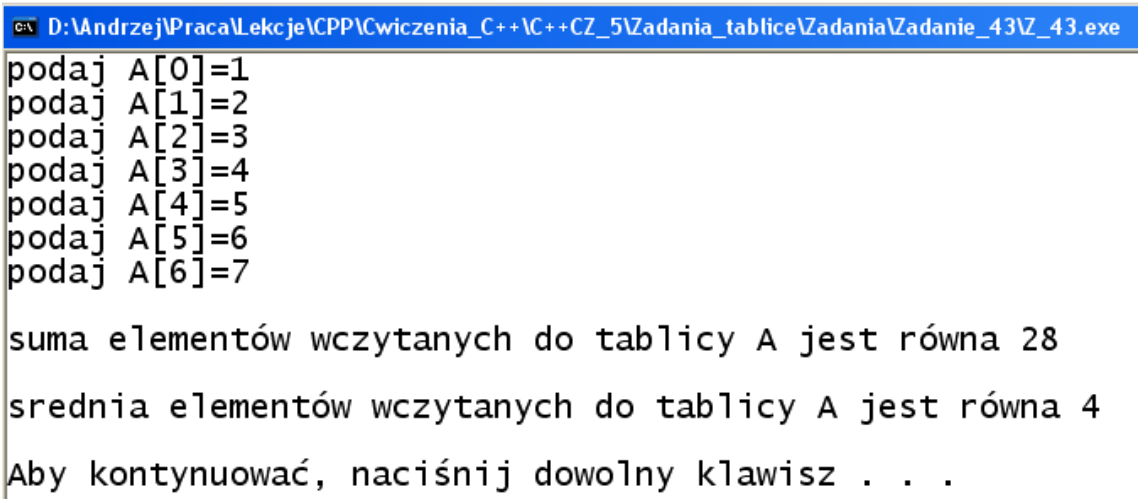
Temat :
Przykład demonstruje wczytywanie do tablicy siedmioelementowej jednowymiarowej oraz wypisanie wczytanych wartości.

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
  int a[7];
  int i;
   for( i=0;i<=6;i++)
   {
        cout<<"podaj A["<<i<<"]=";
        cin>>a[i];                          // tu wprowadzamy liczby do kolejnych komórek
   }
 cout<<"w tablicy A jest"<<"\n";
 for(i=0;i<=6;i++)
   {
        cout<<"w komorce A["<<i<<"]="<<a[i]<<"\n"; // tu wypisujemy zawartość komórek

   }
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

```
podaj A[0]=2
podaj A[1]=4
podaj A[2]=5
podaj A[3]=3
podaj A[4]=6
podaj A[5]=7
podaj A[6]=8
w tablicy A jest
w komorce A[0]=2
w komorce A[1]=4
w komorce A[2]=5
w komorce A[3]=3
w komorce A[4]=6
w komorce A[5]=7
w komorce A[6]=8
Aby kontynuować, naciśnij dowolny klawisz . . .
```

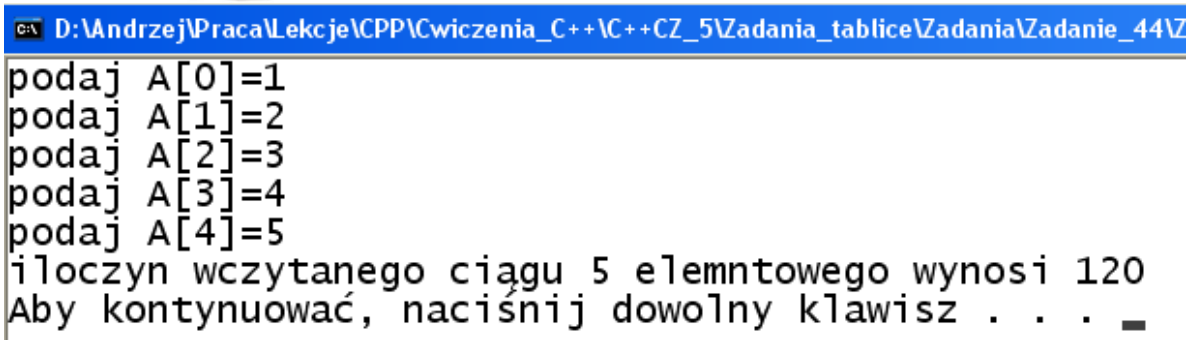
ZADANIE 43

Napisać program obliczający sumę oraz jego średnia arytmetyczną wczytanego ciągu do tablicy siedmioelementowej. Wykonaj schemat blokowy.
Wskazówka: Wykorzystaj treść przykładu poprzedniego.



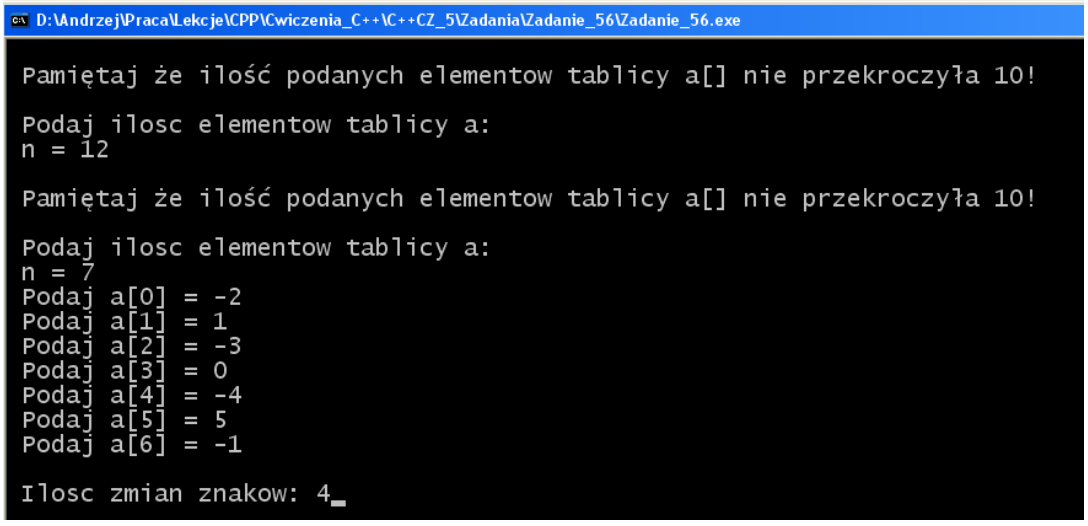
ZADANIE 44

Napisać program obliczający iloczyn wczytanego ciągu do tablicy pięcioelementowej.
Wykonaj schemat blokowy.
Wskazówka: Zastanów się, jaką wartość będzie miała zmienna ILOCZYN przed przystąpieniem do obliczania iloczynu.



ZADANIE 56

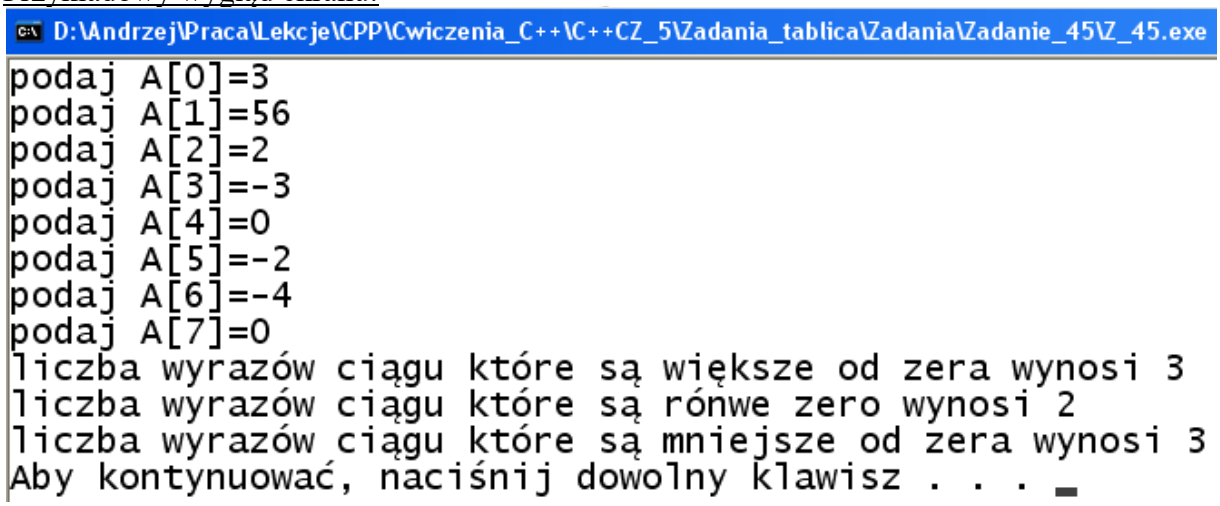
Napisać program wyznaczający ilości zmian znaków w tablicy składającej się maksymalnie z 10 elementów.



ZADANIE 46

Napisać program zliczający ilość wyrazów ciągu które:
są większe od zera,
równe zero,
mniejsze od zera
dla ciągu wczytanego do tablicy ośmioelementowej. Wykonaj schemat blokowy.
Wskazówka:
Utwórz trzy liczniki: L_mniejsze, L_wieksze, L_rowne. Nadaj im wartości zero przed procesem zliczania. Powiększaj wartość odpowiedniego licznika o jeden po sprawdzeniu odpowiednich warunków, jaki spełnia każdy element wczytanego ciągu.

Przykładowy wygląd ekranu:



Przykład 27

Wykonaj:
1)Wpisz w zeszycie temat programu.
2)Po rozwiązaniu zapisz listing.

Temat: Program generujący macierz jednostkową 6*6 (siedem wierszy i siedem kolumn). Macierz jednostkowa to tablica liczb, która po przekątnej ma jedynki a pozostałe liczby to zera. Jest to przykład użycia pętli podwójnej oraz tablicy dwuwymiarowej.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>

using namespace std;

int main(int argc, char *argv[])
{
   int a[7][7];
   for(int i=0;i<7;i++)
   {
         for(int j=0;j<7;j++)
         {
         if(i==j) {a[i][j]=1;}
```

```
        else
        a[i][j]=0;
        }
  }                    // pierwsza pętla wstawia wartości 0 lub1

  for(int i1=0;i1<7;i1++)
   {
        for(int j1=0;j1<7;j1++)
        {
         cout<<a[i1][j1]<<" ";
        }
      cout<<"\n";      // tutaj uzyskujemy przejście do wiersza poniżej
   }                   // druga pętla wypisuje macierz

  system("PAUSE");
  return EXIT_SUCCESS;
}
```

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Zadania_tablica\Przyklady\P_27\P_27.exe
1 0 0 0 0 0 0
0 1 0 0 0 0 0
0 0 1 0 0 0 0
0 0 0 1 0 0 0
0 0 0 0 1 0 0
0 0 0 0 0 1 0
0 0 0 0 0 0 1
Aby kontynuować, naciśnij dowolny klawisz . . .
```

ZADANIE 47

Napisać program generowania następującej tablicy 5*5. Do generowania tablicy użyj tablic i pętli.

```
9 1 1 1 1
1 9 1 1 1
1 1 9 1 1
1 1 1 9 1
1 1 1 1 9
```

Wydrukuj tablicę wierszami .

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Zadania_tablica\Zadania\Zadanie_47\Z_47.exe
9 1 1 1 1 Aby kontynuować, naciśnij dowolny klawisz . . .
1 9 1 1 1 Aby kontynuować, naciśnij dowolny klawisz . . .
1 1 9 1 1 Aby kontynuować, naciśnij dowolny klawisz . . .
1 1 1 9 1 Aby kontynuować, naciśnij dowolny klawisz . . .
1 1 1 1 9 Aby kontynuować, naciśnij dowolny klawisz . . .
Aby kontynuować, naciśnij dowolny klawisz . . .
```

Przykład 28

Wykonaj:

1)Wpisz w zeszycie temat programu.

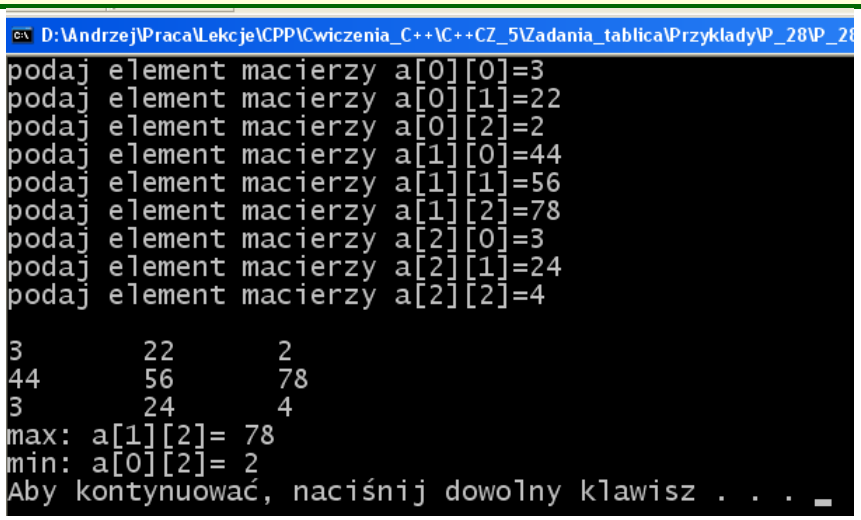
2)Po rozwiązaniu zapisz listing.

3)Narysuj schemat blokowy.

Temat: Program wczytujący macierz 3 na 3. Macierz wypisywana jest wierszami. Znajdowany jest element **maksymalny i minimalny**. Wypisywany jest element minimalny oraz maksymalny wraz z miejscem w tablicy.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>

using namespace std;

int main(int argc, char *argv[])
{
int macierz[3][3]; // deklaracja tablicy
// wczytywanie macierzy
  for(int i=0;i<3;i++)
    {
          for(int j=0;j<3;j++)
            {
              cout << "podaj element macierzy a"<<"[" <<i<<"][" << j<<"]=";
              cin>> macierz[i][j];
             }
    }
cout << "\n";   // nowy wiersz
     for(int ii=0;ii<3;ii++)
      {
         for(int ji=0;ji<3;ji++)
           {
              cout << macierz[ii][ji]<<"\t";
            }
              cout << "\n";
      }
int max=macierz[0][0]; // nadanie wartości początkowej elementowi maksymalnemu "max" -
//tutaj pierwsza wartość w tablicy
int min=macierz[0][0]; // nadanie wartości początkowej elementowi minimalnemu "min"-
//tutaj również pierwsza wartość w tablicy
int minx=0;  // zerowanie wartości początkowej dla indeksów końcowych tabeli dla max i min
int miny=0;
int maxx=0;
int maxy=0;
   for(int z=0;z<3;z++)
    {
     for(int x=0;x<3;x++)
      {
       if (macierz[z][x]<min) // szukam punktu o min wartości
         {
          min=macierz[z][x]; // tu podstawiam znalezioną wartość punktu min
          minx=z;           //wstawiam jego współrzędne do końcowej wartości min
          miny=x;
```

```
        }
      if (macierz[z][x]>max) // szukam punktu o max wartości
        {
         max=macierz[z][x];
         maxx=z;
         maxy=x;
        }
       }
     }
   cout << "max: a["<<maxx<< "]["  <<maxy <<"]= "<< max
   << "\nmin: a["<<minx << "]["<< miny <<"]= " << min;
   cout<<"\n";
   system("PAUSE");
   return EXIT_SUCCESS;
}
```



ZADANIE 48

Napisz program wczytujący macierz 5 na 2. Podczas wczytywania wyświetlany jest indeks wczytywanego elementu.

- Macierz wypisywana jest wierszami.
- W rezultacie końcowym do macierzy jednowymiarowej A o pięciu elementach zapisz średnie arytmetyczne wierszy – wypisz elementy tablicy A.
- Znajdź element maksymalny i minimalny macierzy (tablicy) A. Wypisz element minimalny oraz maksymalny wraz z miejscem w tablicy.

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Zadania_tablica\Zadania\Zadanie_48\Zadanie_48.exe
Podaj element a[0] [0] = 4
Podaj element a[0] [1] = 6
Podaj element a[1] [0] = 46
Podaj element a[1] [1] = 5
Podaj element a[2] [0] = 3
Podaj element a[2] [1] = 56
Podaj element a[3] [0] = 4
Podaj element a[3] [1] = 3
Podaj element a[4] [0] = 24
Podaj element a[4] [1] = 5
4 6     srednia w wierszu 0 macierzy wynosi     5

46 5    srednia w wierszu 1 macierzy wynosi     25.5

3 56    srednia w wierszu 2 macierzy wynosi     29.5

4 3     srednia w wierszu 3 macierzy wynosi     3.5

24 5    srednia w wierszu 4 macierzy wynosi     14.5

tablica srednich wygląda nastepująco:   A[ 5 25.5 29.5 3.5 14.5 ]

max: A[2]= 29.5
min: A[3]= 3.5
Aby kontynuować, naciśnij dowolny klawisz . . .
```

ZADANIE 49

Napisz program wczytujący macierz 3 na 4. Podczas wczytywania zapisany jest indeks wczytywanego elementu.

- Macierz po wczytaniu wypisywana jest wierszami.
- Znajdowana jest suma wszystkich elementów macierzy.
- Komputer pyta się o liczbę i następuje mnożenie macierzy przez liczbę. Po pomnożeniu wyświetlania jest wierszami tablica wynikowa.

- **Przykład**
- Dane jest macierz **A** oraz liczba **k**, przez którą należy macierz **A** przemnożyć.
- Dane:

$$A = \begin{bmatrix} 2 & 7 & 4 \\ 5 & 2 & 1 \\ 6 & 3 & 4 \end{bmatrix}; k = 5$$

- Rozwiązanie:

$$A \cdot k = \begin{bmatrix} 2 & 7 & 4 \\ 5 & 2 & 1 \\ 6 & 3 & 4 \end{bmatrix} \cdot 5 = \begin{bmatrix} 10 & 35 & 20 \\ 25 & 10 & 5 \\ 30 & 15 & 20 \end{bmatrix}$$

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Zadania_tablica\Zadania\Zadanie_49\Zad
Podaj element a[0] [0] = 3
Podaj element a[0] [1] = 2
Podaj element a[0] [2] = 4
Podaj element a[0] [3] = 1
Podaj element a[1] [0] = 5
Podaj element a[1] [1] = 6
Podaj element a[1] [2] = 3
Podaj element a[1] [3] = 5
Podaj element a[2] [0] = 10
Podaj element a[2] [1] = 11
Podaj element a[2] [2] = 2
Podaj element a[2] [3] = 4
3       2       4       1
5       6       3       5
10      11      2       4
suma wszystkich elementów macierzy a= 56
Podaj liczbę przez którą chcesz pomnożyć macierz: 3
9       6       12      3
15      18      9       15
30      33      6       12
```

ZADANIE 50

Napisz program obliczający wyznacznik 3x3 metodą Sarussa.

- Wczytaj macierz 3x3
- Skopiuj potrzebne do obliczeń elementy macierz i utwórz nową macierz 3x5
- Nową macierz 3x5 wypisz na ekranie
- Podaj wartość wyznacznika.

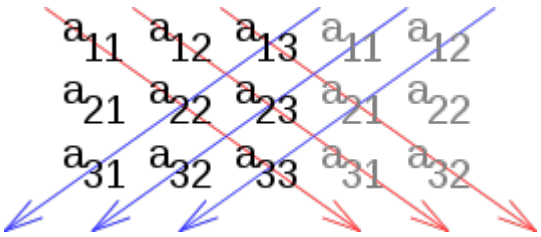
Do kopiowania macierzy, wypisywania macierzy oraz wyliczeń iloczynów składowych wzoru na wyznacznik **użyj tablic oraz pętli FOR.**

Reguła Sarrusa, albo **schemat Sarrusa** to praktyczny sposób obliczania wyznacznika stopnia 3. Algorytm ten został odkryty przez francuskiego matematyka Pierre'a Sarrusa.

Aby obliczyć wyznacznik:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

dopisuje się z jego prawej strony dwie pierwsze kolumny,



oblicza sumę iloczynów wzdłuż "czerwonych strzałek" i odejmuje od niej sumę iloczynów wzdłuż "niebieskich strzałek".

W innej wersji schematu dopisuje się dwa pierwsze wiersze **pod** wyznacznikiem, a następnie postępuje jak wyżej.

Ogólny wzór ma postać następującą:

$$(a_{11}\cdot a_{22}\cdot a_{33}+a_{12}\cdot a_{23}\cdot a_{31}+a_{13}\cdot a_{21}\cdot a_{32})-(a_{13}\cdot a_{22}\cdot a_{31}+a_{11}\cdot a_{23}\cdot a_{32}+a_{12}\cdot a_{21}\cdot a_{33})$$

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Zadania_tablica\Zadania\Zadanie_50\Zadanie_50.exe
Podaj element tablicy a[0] [0]= 3
Podaj element tablicy a[0] [1]= 2
Podaj element tablicy a[0] [2]= 4
Podaj element tablicy a[1] [0]= 0
Podaj element tablicy a[1] [1]= 6
Podaj element tablicy a[1] [2]= 1
Podaj element tablicy a[2] [0]= 4
Podaj element tablicy a[2] [1]= 9
Podaj element tablicy a[2] [2]= 7

Pełna macierz ze skopiowanymi elementami wyglada następująco:
3 2 4 3 2
0 6 1 0 6
4 9 7 4 9

Wyznacznik tablicy a = (126+8+0)-(96+27+0)=11
Aby kontynuować, naciśnij dowolny klawisz . . .
```

ZADANIE 51

Napisz program odwracający wczytaną tablicę jednoelementową. Podczas wczytywania zapisywany jest indeks wczytywanego elementu.

- Maksymalna wielkość tablicy 10 elementów;
- Program pyta o liczbę elementów jakie wprowadzimy do tablicy;
- Zapewnij kontrolę wprowadzenia ilość elementów oraz w przypadku przekroczenia przedziału, podaj stosowny komunikat;

W trakcie rozwiązywania zadania wykorzystaj zmienną **schowek** do przechowywania danej, podobną metodę będziesz stosował w algorytmach sortowania.

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Zadania\Zadanie_51\main.exe

Podaj ilosc elementow tablicy a:
n = 11
Tablica musi zawierać minimum 1 a maksymum 10 elementów !
Podaj ilosc elementow tablicy a:
n = 5
Podaj a[0] = 3
Podaj a[1] = 5
Podaj a[2] = 7
Podaj a[3] = 8
Podaj a[4] = 0

Odwrocona tablica a:
a[0] = 0
a[1] = 8
a[2] = 7
a[3] = 5
a[4] = 3
```

Sortowanie

Ćwiczenia wykonane pisemnie (notatki w zeszycie)

Zapisz numer pytania (przed pytaniem np. Pytanie 1) poniżej zapisz treść pytania. Treść pytania podkreśl na **zielono** a pod treścią pytania zapisz odpowiedź.

Pytanie 1
Podaj definicję sortowania;
Pytanie 2
Wymień metody sortowania;
Pytanie 3
Podaj opis algorytmu sortowania metodą bąbelkową
Pytanie 4
Przepisz do zeszytu przykład sortowania metodą bąbelkową.
Pytanie 5
Podaj opis algorytmu sortowania metodą przez wybór
Pytanie 6
Przepisz do zeszytu przykład sortowania metodą przez wybór.
Pytanie 7
Podaj opis algorytmu sortowania metodą przez wstawianie
Pytanie 8
Przepisz do zeszytu przykład sortowania metodą przez wstawianie.

Sortowanie — oznacza proces porządkowania według pewnego klucza np.: od największego do najmniejszego, najmniejszego do największego, alfabetycznie itp.
Sortowanie może odbywać się według wielu różnych algorytmów, różniących się ilością wykonywanych operacji oraz szybkością działania.

Metody sortowania.

- Sortowanie bąbelkowe (angielskie bubble sort),
- Sortowanie bąbelkowe z kresem górnym
- Sortowanie przez wybór (angielskie selectsort)
- Sortowanie przez wstawianie (insertion sort)
- Sortowanie przez zliczanie (counting sort)
- Sortowanie pozycyjne (radix sort)
- Sortowanie wyrazów
- Sortowanie QuickSort

Opis do algorytmu sortowania metodą bąbelkową

Metoda **bąbelkowa** polega: (porządkowanie od najmniejszego do największego – lub odwrotnie), ustawiamy się na pierwszym wyrazie ciągu i porównujemy go z drugim jeśli pierwszy jest większy od drugiego to zamieniamy je miejscami, następnie umieszczamy się na drugim i porównujemy z trzecim dokonując przestawienia lub nie. Porównując parami dochodzimy do końca ciągu (ustawiamy się na przedostatnim wyrazie). Jeśli podczas przechodzenia nastąpiła zmiana to musimy to zapamiętać jeśli nie to znaczy, że ciąg jest ustawiony prawidłowo. Po przejściu całego ciągu ustawiamy się ponownie na początek i sprawdzamy czy w poprzednim przechodzeniu było przestawienie jeśli nie było zamian to kończymy program jeśli było przestawienie to proces powtarzamy tyle razy aż nie będzie przestawienia.

Tablicę uporządkować w porządku rosnącym.

9	2	7	10	8	4

przebieg porządkowania będzie następujący

pierwszy przebieg zamienione elementy
krok 1 2 9 7 10 8 4 9 2
krok 2 2 7 9 10 8 4 9 7
krok 3 2 7 9 8 10 4 10 8
krok 4 2 7 9 8 4 10 10 4
drugi przebieg zamienione elementy
krok 1 2 7 8 9 4 10 9 8
krok 2 2 7 8 4 9 10 9 4

trzeci przebieg zamienione elementy
krok 1 2 7 4 8 9 10 8 4
czwarty przebieg zamienione elementy
krok 1 2 4 7 8 9 10 7 4
w piątym przebiegu nie nastąpiła zmiana tzn., że można skończyć sortowanie.

ZADANIE 52
Treść praktyczna zadania
Temat: Dokonaj sortowania tablicy jednowymiarowej metodą bąbelkową z użyciem listy kroków, która jest zapisana poniżej.

Rozwiązanie:
1)Przepisz do zeszytu listę kroków dla sortowania bąbelkowego.
2)Przepisz wzór do przestawiania elementów w ciągu (ten ze schowkiem).
3)Utwórz tablicę t_trzy_pierwsze_litery_nazwiska np. t_kow o siedmiu elementach.

Nadaj elementom tablicy następujące wartości:
t_kow[0]= numer_z_dziennika;
t_kow[1]= liczba_liter_imienia;
t_kow[2]= liczba_liter_nazwiska;

t_kow[3]= –miesiąc_urodzenia;
t_kow[4]= dzień_urodzenia;
t_kow[5]= numer_buta
t_kow[6]= wag_w_kg

4)Wyświetl tablicę t_kow

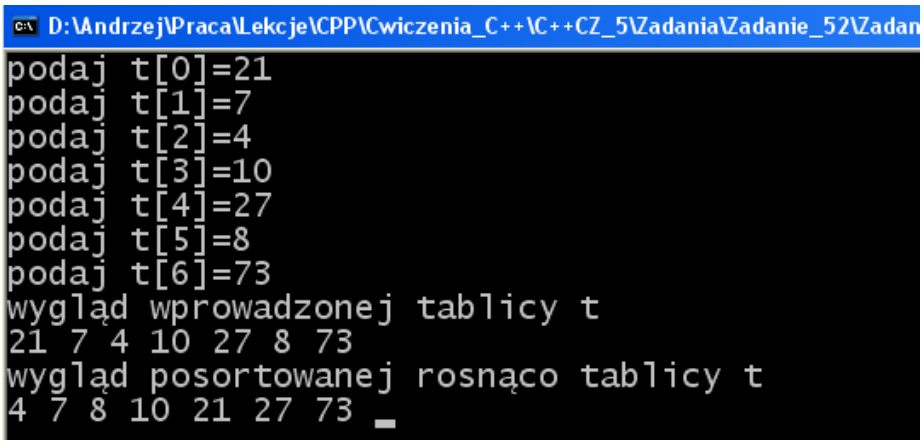
5)Dokonaj sortowania wg listy kroków
Użyj zmiennej SCHOWEK_trzy_pierwsze_litery. Użyj dwóch pętli.
Algorytmy mogą być przedstawione z użyciem listy kroków. Poniższa lista kroków przedstawia sortowanie metodą bąbelkową.

KROK1: Sprawdzaj zmianę dopóki, nie będzie przestawień liczb w pętli, w tym celu utwórz zmienną zmiana =0;
KROK2: Dla i = 0,1,…,n-1: jeśli t[i]>t[i+1], to t[i]←t[i+1]
KROK3: Zakończ algorytm.

Opis:
Zapis oznacza t[i]←t[i+1], że należy zamienić miejscami elementy t[i] i t[i+1]. Zamianę można uzyskać poprzez:
SCHOWEK=t[i];
t[i]=t[i+1];
t[i+1]=SCHOWEK;
zamiana=zamiana+1;
gdzie SCHOWEK jest zmienną pomocniczą a t[i] jest aktualnym elementem.
Pamiętaj, że pierwszy element tablicy to zero np. t[0]. Musisz uwzględnić ten fakt w konstrukcji pętli tak, aby zacząć od elementu zero czyli wartość początkowa zmiennej sterującej pętlą ma wartość zero. Musisz pomyśleć również, kiedy pętla ma przestać się wykonywać.

6)Wyświetl tablicę po sortowaniu.

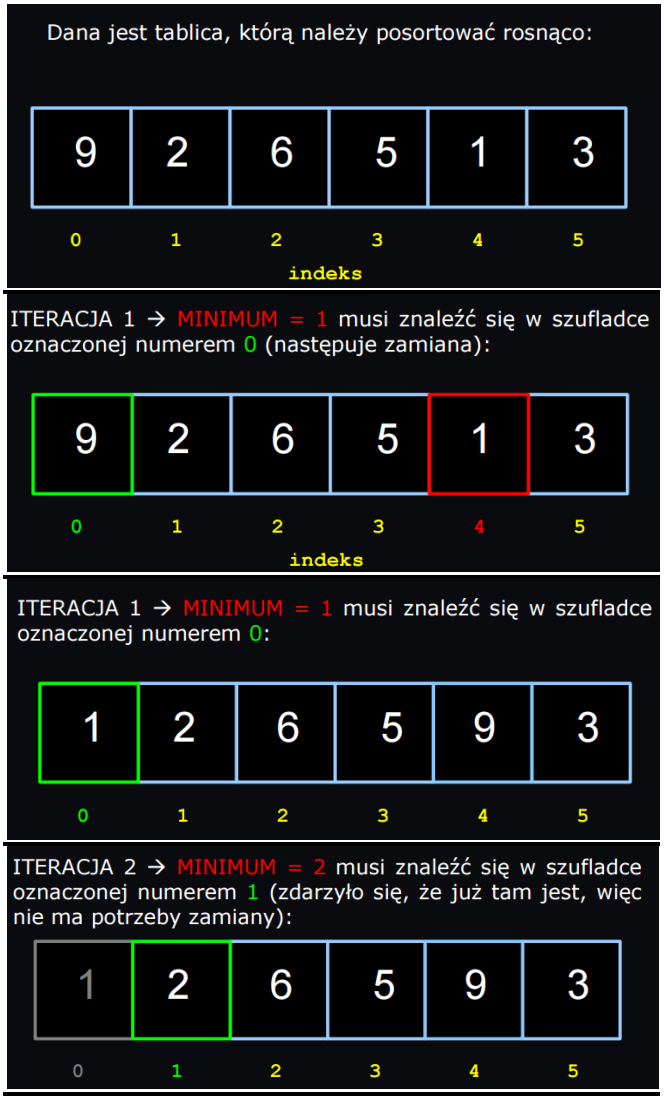
Koniec zadania 52

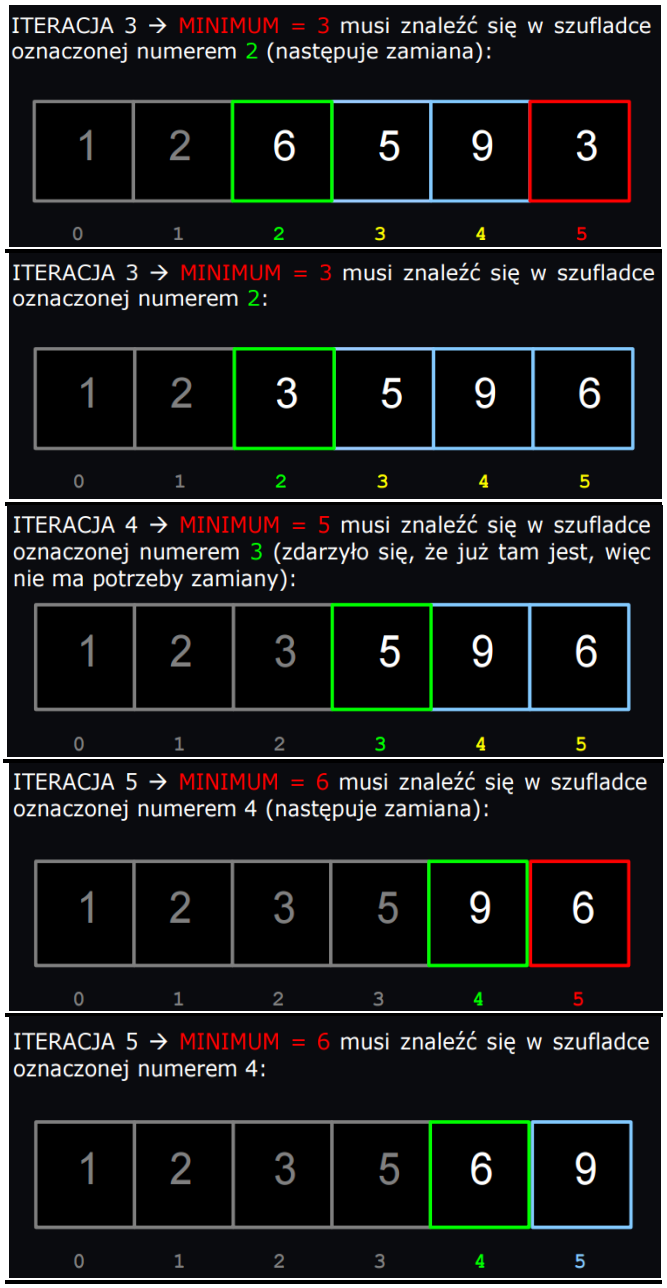


Opis algorytmu sortowania przez wybór

Sortowanie to polega na iteracyjnym znajdowaniu najmniejszego (w sortowaniu rosnącym) lub największego (w sortowaniu malejącym) elementu i zamianie go z pierwszym elementem w tablicy, po czym rozmiar tablicy zmniejszamy o jeden.
ALGORYTM:
a) wyznaczyć najmniejszy element w tablicy [0..n],
b) zamienić go miejscami z zerowym elementem tablicy,
c) wyznaczyć najmniejszy element w tablicy [1..n],
d) zamienić go miejscami z pierwszym elementem tablicy,
e) wyznaczyć najmniejszy element w tablicy [2..n],
f) zamienić go miejscami z pierwszym elementem tablicy,

itd. aż do posortowania całej tablicy ☺





ZADANIE 53

Treść praktyczna zadania

Temat: Dokonaj sortowania tablicy jednowymiarowej metodą sortowania przez wybór z użyciem listy kroków lub według algorytmu sortowania przez wybór:

Rozwiązanie:

1)Przepisz do zeszytu listę kroków dla sortowania przez wybór.
2)Przepisz wzór do przestawiania elementów w ciągu (ten z buforem).
3)Utwórz tablicę tab_trzy_pierwsze_litery_nazwiska np. tab_kow o siedmiu elementach.

Nadaj elementom tablicy następujące wartości:
tab_kow[0]= numer_z_dziennika;
tab_kow[1]= liczba_liter_imienia;
tab_kow[2]= Twój wzrost;

tab_kow[3]= liczba_liter_nazwiska;
tab_kow[3]= ostatnie dwie cyfry obecnego roku;
tab_kow[5]= dzisiejszy dzień;
tab_kow[6]= obecny miesiąc;

4)Wyświetl tablicę tab_kow

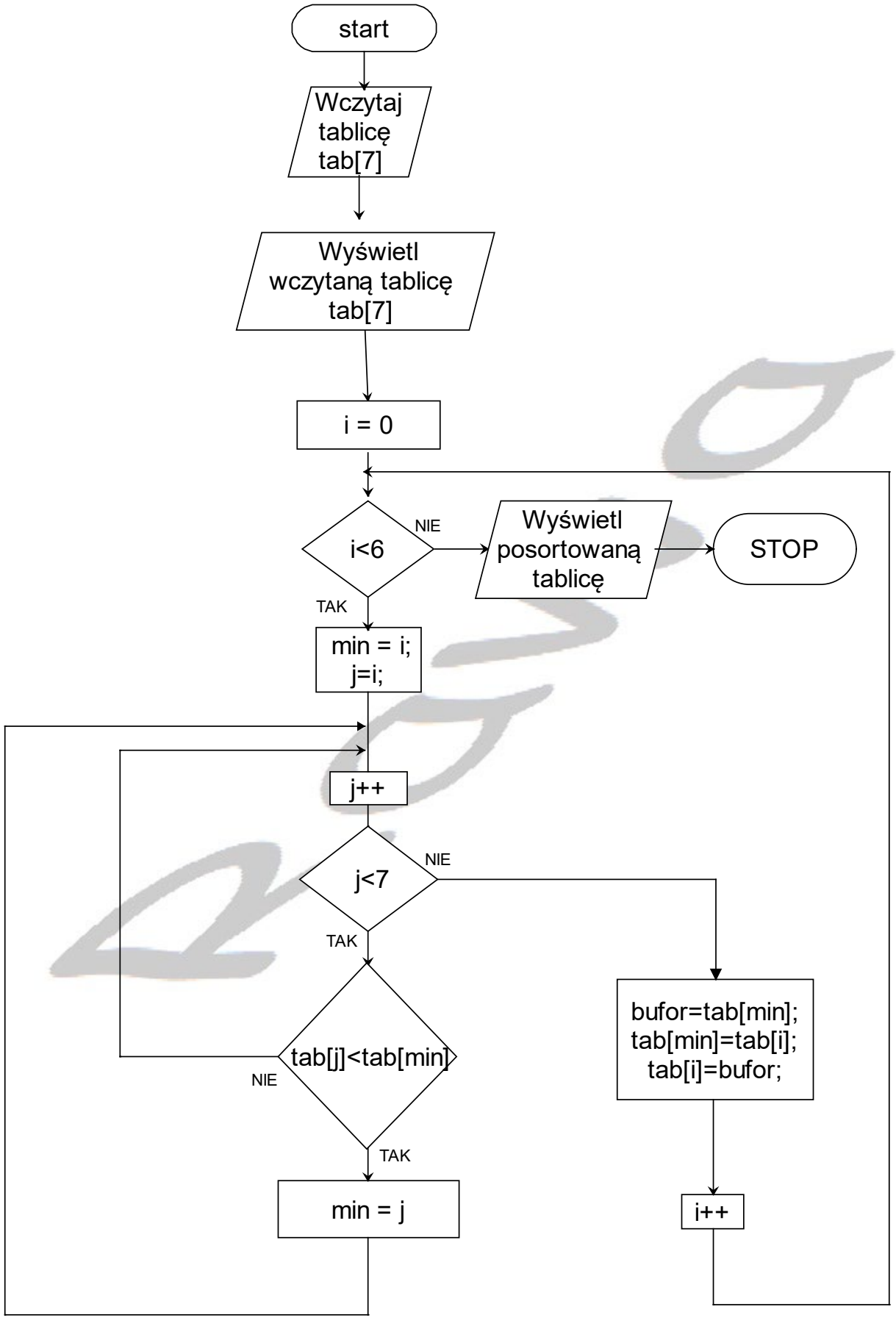
5)Dokonaj sortowania wg listy kroków lub algorytmu przedstawionego poniżej.

Użyj zmiennej Bufor_trzy_pierwsze_litery. Użyj dwóch pętli .
Algorytmy mogą być przedstawione z użyciem listy kroków. Poniższa lista kroków przedstawia sortowanie metodą przez wybór.

KROK1: W pierwszej pętli ustawiam obecnie wpisany na pozycję pierwszą element tablicy jako wartość minimum, czyli dla i = 0,1,…,n-2; min=i
KROK2: Dla j = i+1,1,…,n-1: jeśli tab[j]<tab[min]); // sprawdzam np. element tab[1] z tab[0] min=j;
KROK3: wykonuję zamianę elementów tablicy miejscami:
bufor=tab[min]; // wstawiam min element tablicy do bufora np. bufor=tab[3]
tab[min]=tab[i]; // ustawiam kolejny element tablicy np. tab[3]=tab[0]
tab[i]=bufor; // ustawiam tab[0]=tab[3]

Pamiętaj, że pierwszy element tablicy to zero np. tab[0]. Musisz uwzględnić ten fakt w konstrukcji pętli tak, aby zacząć od elementu zero czyli wartość początkowa zmiennej sterującej pętlą ma wartość zero.

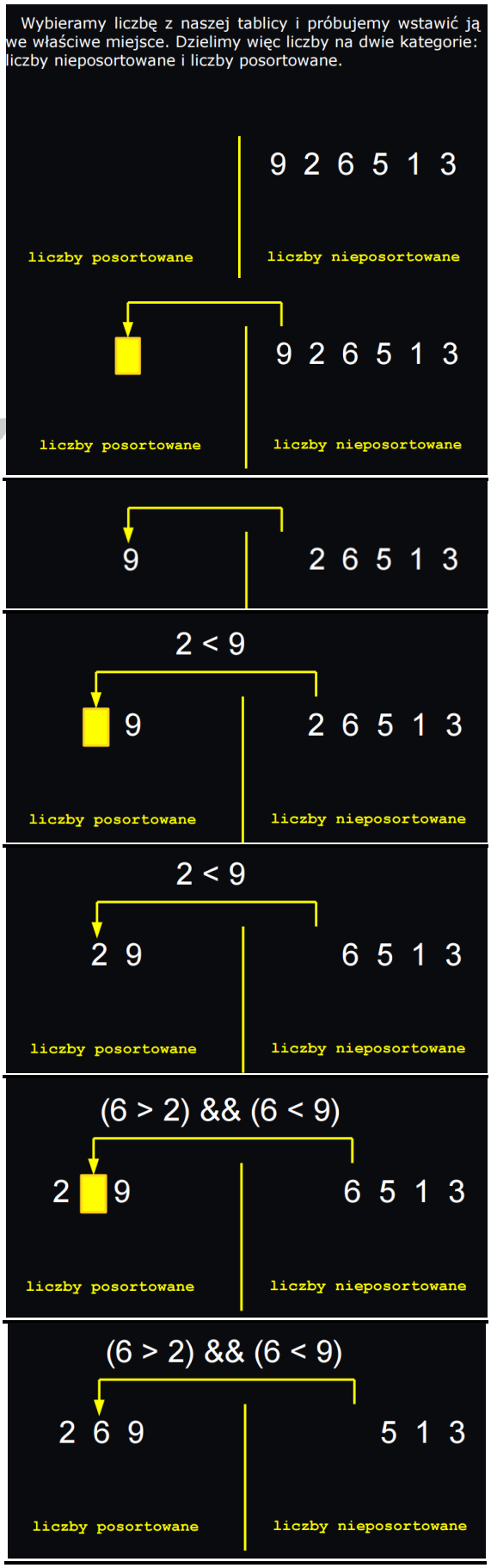
6)Wyświetl tablicę po sortowaniu.
Koniec zadania 53

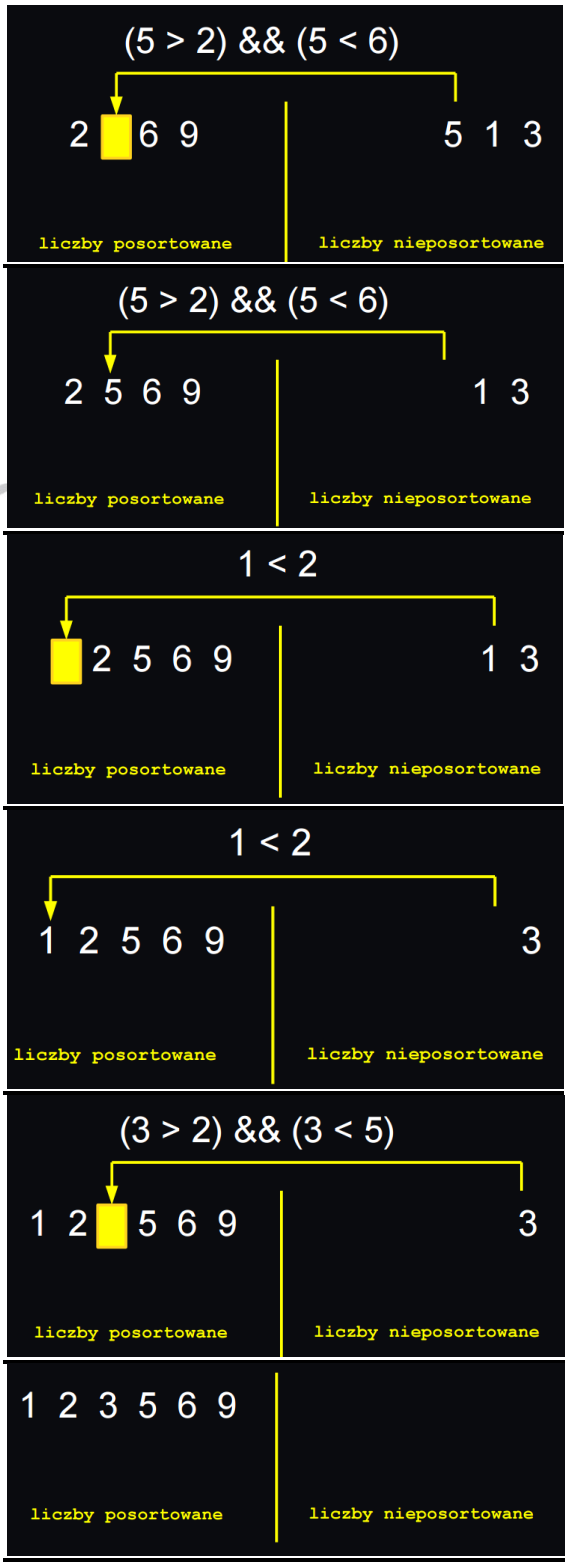

start
Wczytaj tablicę tab[7]
Wyświetl wczytaną tablicę tab[7]
i = 0
i<6
NIE
Wyświetl posortowaną tablicę
STOP
TAK
min = i;
j=i;
j++
j<7
NIE
TAK
tab[j]<tab[min]
NIE
TAK
min = j
bufor=tab[min];
tab[min]=tab[i];
tab[i]=bufor;
i++

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Zadania\Zadanie_53\Zadanie_53.exe
Podaj element tablicy: [0] = 21
Podaj element tablicy: [1] = 4
Podaj element tablicy: [2] = 179
Podaj element tablicy: [3] = 7
Podaj element tablicy: [4] = 15
Podaj element tablicy: [5] = 25
Podaj element tablicy: [6] = 9

wygląd wprowadzonej tablicy tab [ 21 4 179 7 15 25 9  ]

posortowana tablica wygląda następująco: tab[ 4 7 9 15 21 25 179  ]

Aby kontynuować, naciśnij dowolny klawisz . . .
```

Opis algorytmu sortowania przez wstawianie

Zasada działania tego algorytmu jest często porównywana do porządkowania kart w wachlarz podczas gry. Każdą kartę wstawiamy w odpowiednie miejsce, tzn. po młodszej, ale przed starszą. Podobnie jest z liczbami. Pierwszy element pozostaje na swoim miejscu. Następnie bierzemy drugi i sprawdzamy, w jakiej relacji jest on z pierwszym. Jeśli jest niemniejszy, to zostaje na drugim miejscu, w przeciwnym wypadku wędruje na pierwsze miejsce. Dalej sprawdzamy trzeci element (porównujemy go do dwóch pierwszych i wstawiamy w odpowiednie miejsce), czwarty (porównujemy z trzema pierwszymi), piąty itd. Idea działania algorytmu opiera się na podziale ciągu na dwie części: pierwsza jest posortowana, druga jeszcze nie. Wybieramy kolejną liczbę z drugiej części i wstawiamy ją do pierwszej. Ponieważ jest ona posortowana, to szukamy dla naszej liczby takiego miejsca, aby liczba na lewo była niewiększa a liczba na prawo niemniejsza.


Wybieramy liczbę z naszej tablicy i próbujemy wstawić ją we właściwe miejsce. Dzielimy więc liczby na dwie kategorie: liczby nieposortowane i liczby posortowane.
9 2 6 5 1 3
liczby posortowane
liczby nieposortowane
9 2 6 5 1 3
liczby posortowane
liczby nieposortowane
9
2 6 5 1 3
2 < 9
9
2 6 5 1 3
liczby posortowane
liczby nieposortowane
2 < 9
2 9
6 5 1 3
liczby posortowane
liczby nieposortowane
(6 > 2) && (6 < 9)
2 9
6 5 1 3
liczby posortowane
liczby nieposortowane
(6 > 2) && (6 < 9)
2 6 9
5 1 3
liczby posortowane
liczby nieposortowane



ZADANIE 54

Treść praktyczna zadania

Temat: Dokonaj sortowania tablicy jednowymiarowej metodą przez wstawianie na podstawie, algorytmu zapisanego poniżej.

Rozwiązanie:

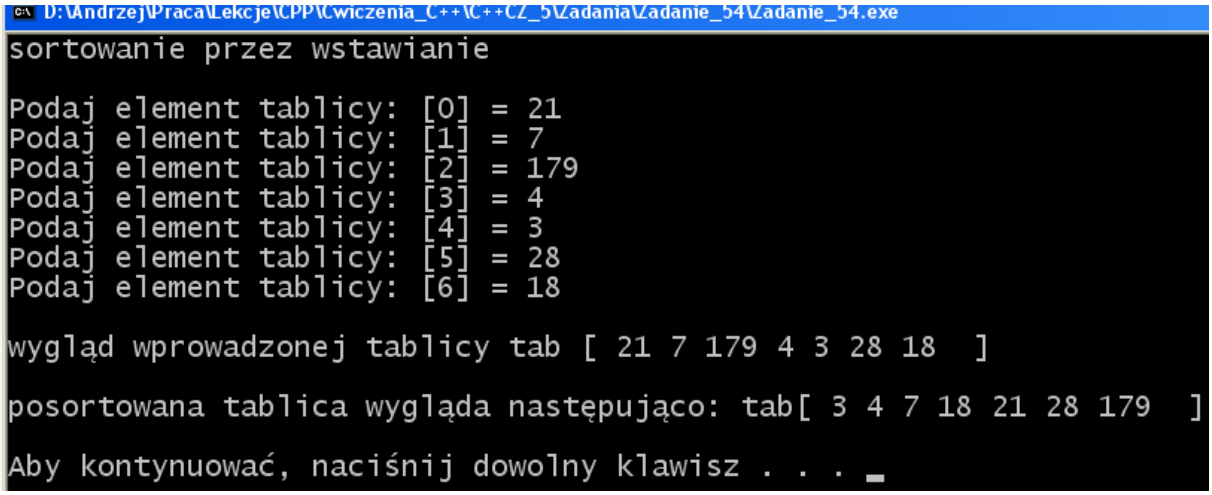
1)Przepisz do zeszytu algorytm dla sortowania przez wstawianie.

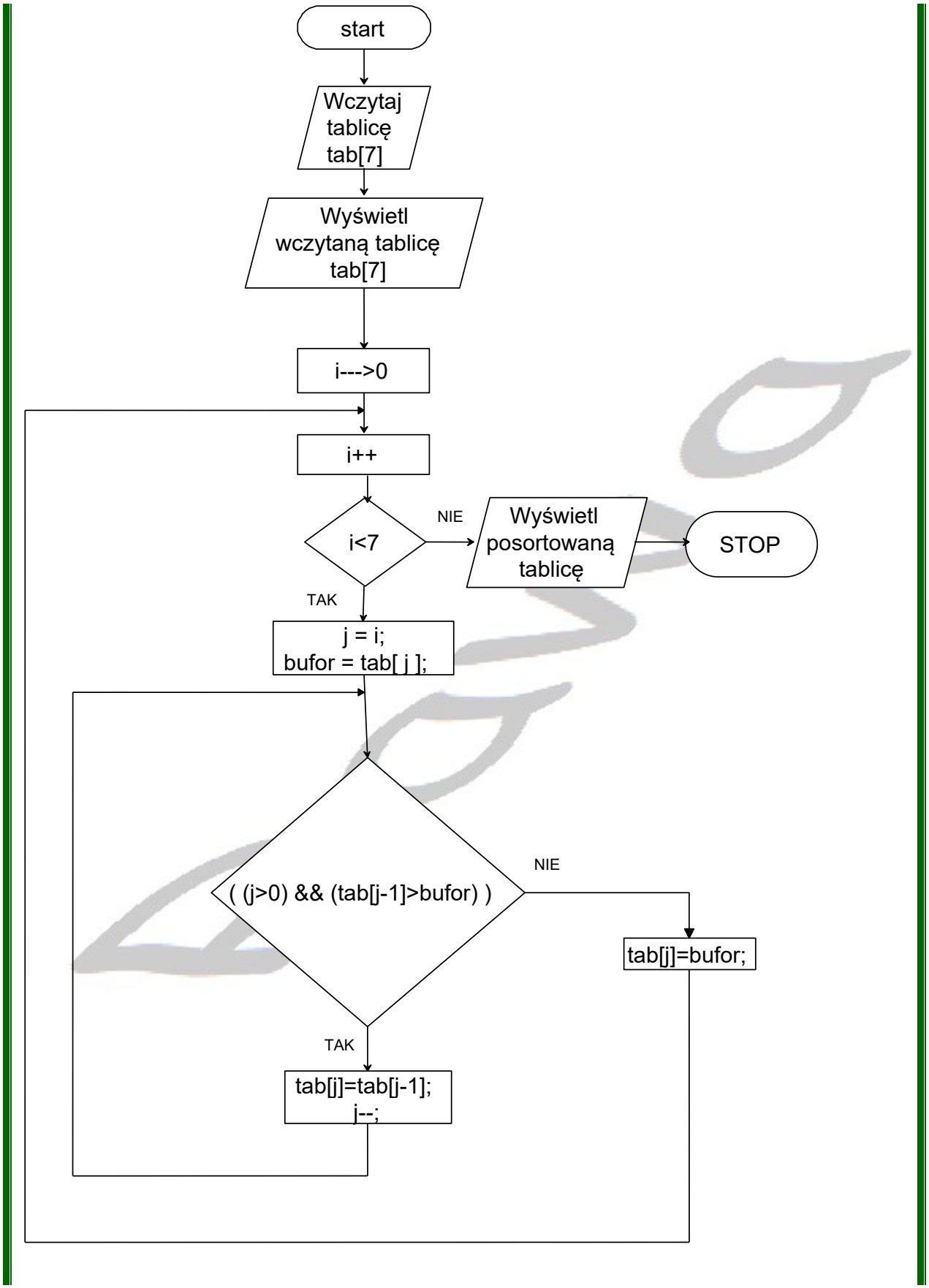
2)Utwórz tablicę tab_trzy_pierwsze_litery_nazwiska np. tab_kow o siedmiu elementach.

Nadaj elementom tablicy następujące wartości:
tab_kow[0]= numer_z_dziennika;
tab_kow[1]= liczba_liter_imienia;
tab_kow[2]= Twój wzrost;
tab_kow[3]= liczba_liter_nazwiska;
tab_kow[3]= numer kolejny dzisiejszej lekcji;
tab_kow[5]= dzisiejszy dzień;
tab_kow[6]= aktualna pełna godzina;

4)Wyświetl tablicę tab_kow

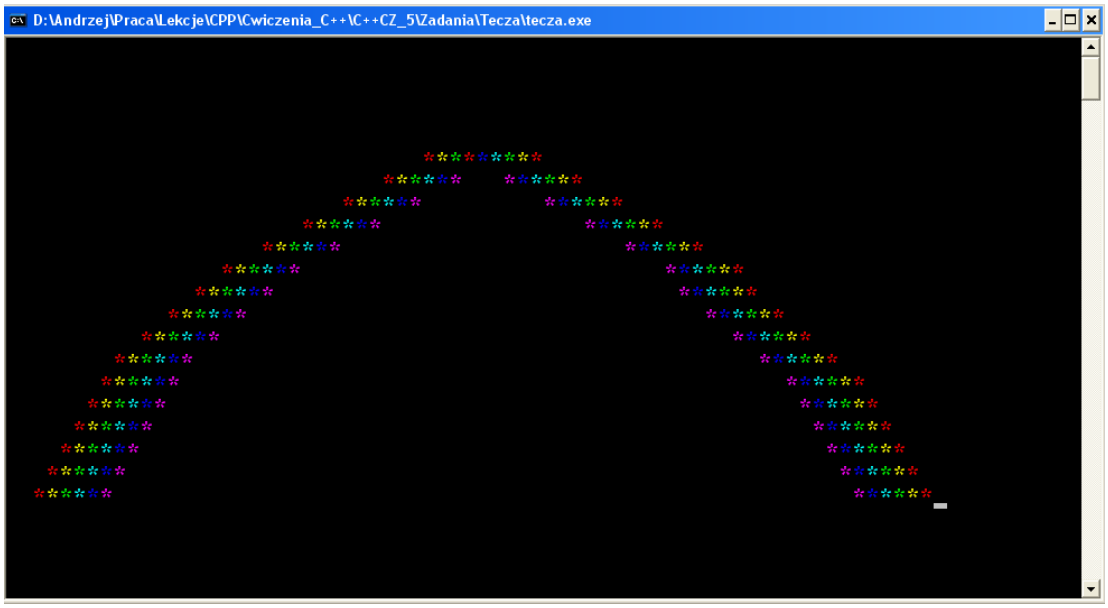
5)Dokonaj sortowania wg algorytmu poniżej:




start
Wczytaj tablicę tab[7]
Wyświetl wczytaną tablicę tab[7]
i--->0
i++
i<7
NIE
Wyświetl posortowaną tablicę
STOP
TAK
j = i;
bufor = tab[j];
((j>0) && (tab[j-1]>bufor))
NIE
tab[j]=bufor;
TAK
tab[j]=tab[j-1];
j--;

ZADANIE 55

Napisz program który z użyciem tablic narysuje na ekranie tęczę:
podczas rysowania będą generowane sygnały dźwiękowe,
tęcza będzie powstawała w umiarkowanym tempie od lewej do prawej strony ekranu.



ZADANIE 58 (Skoki narciarskie)

Czesław Kałysz trenuje skoki narciarskie. Jego trener Apollo Tani karmi zawodnika bułeczkami z zawartością bananów oraz skrupulatnie notuje postępy Cześka. Zbliżają się ważne zawody. Czesław weźmie w nich udział, jeśli w wybranym przez trenera okresie jego skoki były coraz dłuższe. Napisz program, który określa, czy Czesław wystartuje w najbliższych zawodach.
Proponuję przyjąć następujące zmienne:
zmienna „skok" - liczba naturalną, która oznacza ilość skoków odnotowanych przez trenera (1 ≤ skok ≤ 1000000); długosc_skoku liczba całkowita z zakresu od 1 do 1000000000 oddzielona pojedynczymi odstępami oznaczające długości skoków Czesia; dwie liczby a i b (0 ≤ a < b < 1000000) oznaczające początek i koniec okresu, w którym badane będą wyniki Czesława. Uwaga! Wyniki Czesia numerujemy od 0.

```
Podaj ilość skoków narciarskich odnotowanych przez trenera:
4
Podaj jakie wyniki skoków uzyskał zawodnik: (liczby rozdziel enterem):
123
134
135
137
Podaj długość sprawdzanego okresu (liczby rozdziel enterem):
0
3
TAK - jedziemy na zawody Aby kontynuować, naciśnij dowolny klawisz . . .
```

```
Podaj ilość skoków narciarskich odnotowanych przez trenera:
5
Podaj jakie wyniki skoków uzyskał zawodnik: (liczby rozdziel enterem):
125
136
134
137
139
Podaj długość sprawdzanego okresu (liczby rozdziel enterem):
0
4
Nic z tego - za mało bananów!
Aby kontynuować, naciśnij dowolny klawisz . . .
```

Ciąg Fibonacciego

definiujemy następująco:

pierwszy i drugi element ciągu jest równy 1. Każdy następny otrzymujemy dodając do siebie dwa poprzednie. Matematycznie wygląda to następująco:

Fn={1 dla n=1; 1 dla dla n=2; Fn−2 +Fn−1 dla n>2}

Inna definicja przedstawia zerowy numer ciągu jako wartość 0, pierwszy jako wartość 1, a każdy następny otrzymujemy dodając dwa poprzednie:

Fn={0 dla n=0; 1 dla dla n=1; Fn−2 +Fn−1 dla n>1}

Dla naszych rozważań przyjmiemy definicję pierwszą.

Kilka kolejnych wyrazów tego ciągu według pierwszej definicji przedstawia się następująco:

1, 1, 2, 3, 5, 8, 13, 21,...

1+1=2 1+2=3 2+3=5 3+5=8 5+8=13 8+13=21

Przykład 29

Napisz program który wypisze nam wartość żądanego elementu ciągu Fibonacciego:

```
#include <cstdlib>
#include <iostream>

using namespace std;

int main(int argc, char *argv[])
{
   int n;
   long double fab[100000]; // wczytywane są długie liczby - typ zmien. o wysok. pojemnosci

   cout << "Podaj który element ciagu chcesz wyświetlić: \n";
   cin>>n;

   fab[0]=1;
   fab[1]=1;

   for (int i=2; i<n; i++)
   {
   fab[i]=fab[i-1]+fab[i-2];  // wyraz kolejny jest sumą dwóch poprzednich liczb
   }                          // wczytywanie liczb Fabonacciego do tablicy
   cout<<n<<" wyraz ciągu Fibonacciego wynosi: " <<fab[n-1]<<"\n";
   // wywołanie konkretnego elementu tablicy Fibonacciego
   system("PAUSE");
   return EXIT_SUCCESS;

}
```

```
D:\Andrzej\Praca\Lekcje\CPP\Cwiczenia_C++\C++CZ_5\Przyklady\P_29\P_29.exe
Podaj który element ciagu chcesz wyświetlić:
8
8 wyraz ciągu Fibonacciego wynosi: 21
Aby kontynuować, naciśnij dowolny klawisz . . .
```

ZADANIE 57

Napisz program który z użyciem tablic wypisze dowolny ciąg liczb Fibonacciego oraz poda dla tych liczb przybliżoną wartość „złotej liczby" – Φ.

```
Podaj który element ciagu chcesz wyświetlić:
15
1 wyraz ciągu Fibonacciego wynosi: 1
2 wyraz ciągu Fibonacciego wynosi: 1
3 wyraz ciągu Fibonacciego wynosi: 2
4 wyraz ciągu Fibonacciego wynosi: 3
5 wyraz ciągu Fibonacciego wynosi: 5
6 wyraz ciągu Fibonacciego wynosi: 8
7 wyraz ciągu Fibonacciego wynosi: 13
8 wyraz ciągu Fibonacciego wynosi: 21
9 wyraz ciągu Fibonacciego wynosi: 34
10 wyraz ciągu Fibonacciego wynosi: 55
11 wyraz ciągu Fibonacciego wynosi: 89
12 wyraz ciągu Fibonacciego wynosi: 144
13 wyraz ciągu Fibonacciego wynosi: 233
14 wyraz ciągu Fibonacciego wynosi: 377
15 wyraz ciągu Fibonacciego wynosi: 610

przybliżenie złotej liczby dla: 15 wyrazu ciągu wynosi: 1.61804
Aby kontynuować, naciśnij dowolny klawisz . . .
```

Użycie składni sort

Funkcja **sort** w C++ jest częścią biblioteki standardowej i służy do sortowania elementów w kontenerze, takim jak tablica czy wektor. Jest zaimplementowana w nagłówku **<algorithm>** i działa na podstawie algorytmu sortowania introspektywnego (ang. *introsort*), który łączy zalety sortowania szybkiego (*quicksort*), sortowania przez kopcowanie (*heapsort*) i sortowania przez wstawianie (*insertion sort*).

Podstawowa składnia:

sort (pierwszy, ostatni);

- pierwszy – iterator wskazujący na początek zakresu do posortowania.
- ostatni – iterator wskazujący na koniec zakresu (za ostatnim elementem).

Przykład SORT

```
#include <iostream>
#include <algorithm>

int main()
{
   int tablica[] = {5, 2, 8, 3, 1};
```

```
    int n = sizeof(tablica) / sizeof(tablica[0]); // Obliczanie liczby elementów w tablicy

    // Sortowanie rosnące
    sort(tablica, tablica + n);

    // Wyświetlanie posortowanej tablicy
    for (int i = 0; i < n; i++) {
        cout << tablica[i] << " ";
    }
    return 0;
}
```

int n = sizeof(tablica) / sizeof(tablica[0]);

Ten zapis służy do obliczania liczby elementów w tablicy statycznej w C++. Rozbijmy to krok po kroku:

1. sizeof(tablica) - funkcja sizeof zwraca rozmiar całej tablicy w bajtach. Na przykład, jeśli tablica zawiera 5 elementów typu int, a jeden int zajmuje 4 bajty, sizeof(tablica) zwróci 20 (5 * 4 bajty).
2. sizeof(tablica[0]) - zwraca rozmiar pierwszego elementu w tablicy. W tym przypadku jest to element typu int, więc zwróci 4 bajty.
3. **Dzielenie:** sizeof(tablica) / sizeof(tablica[0])

   To dzielenie oblicza liczbę elementów w tablicy, dzieląc całkowity rozmiar tablicy przez rozmiar jednego elementu. Wynik tej operacji, zapisany w zmiennej n, będzie oznaczał liczbę elementów w tablicy tablica. (w naszym przykładzie 20 : 4 = 5 elementów w tablicy)

Pamiętaj, że ten sposób działa tylko dla tablic statycznych, których rozmiar jest znany w momencie kompilacji.

sort(tablica, tablica + n);

sort(tablica, tablica + n) Sortuje elementy od pierwszego (adres wskazywany przez tablica) do ostatniego (adres wskazywany przez tablica + n), gdzie n – to liczba elementów w tablicy.

Zadanie SORT
Napisz program który posortuje alfabetycznie nazwiska.
Deklarujemy tablicę stringów o rozmiarze określonym zmienną ROZMIAR a nazwiska wprowadzamy do tablicy string nazwisko dla tak określonej liczby nazwisk.
Sortowania dokonaj przy użyciu funkcji **sort** z biblioteki **<algorithm>.**

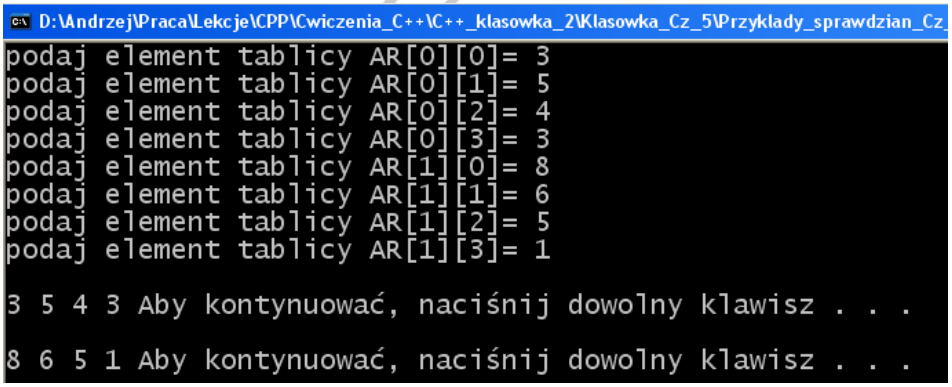
```
Podaj ile nazwisk wprowadzasz do tablicy 5
Podaj 5 nazwiska:
Kowalski
Nowak
Arkoński
Wojciechowski
Szmyt
Nazwiska posortowane alfabetycznie:
Arkoński
Kowalski
Nowak
Szmyt
Wojciechowski
```

PRZYKŁADOWE ZADANIA NA SPRAWDZIAN

ZADANIE 1

Napisz program, który zarezerwuje tablicę 2*4.

- Do tablicy wczytaj z klawiatury kolejne liczby naturalne wierszami.
- Wypisz zawartość tablicy po wczytaniu danych wierszami.
- Nazwa tablicy to Twoje inicjały(bez polskich liter).

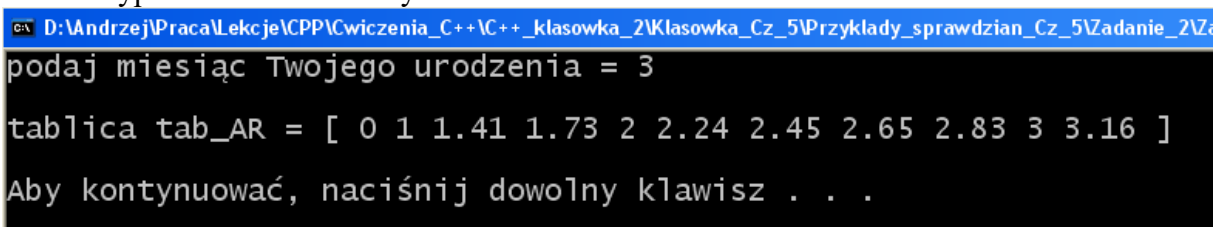


ZADANIE 2

Zarezerwuj tablicę:

- jednowymiarową,
- nazwa tablicy: tab_dwie_pierwsze litery nazwiska ucznia,

- wymiar tablicy: 8+miesiąc Twojego urodzenia (program pyta o miesiąc),
- zawartości tablicy to pierwiastki z kolejnych liczb naturalnych:
  tab[0]=0
  tab[1]=1
  tab[2]=1.41
  ……
  czyli tab[i]=pierwastek(i)
- wypisz zawartość tablicy.

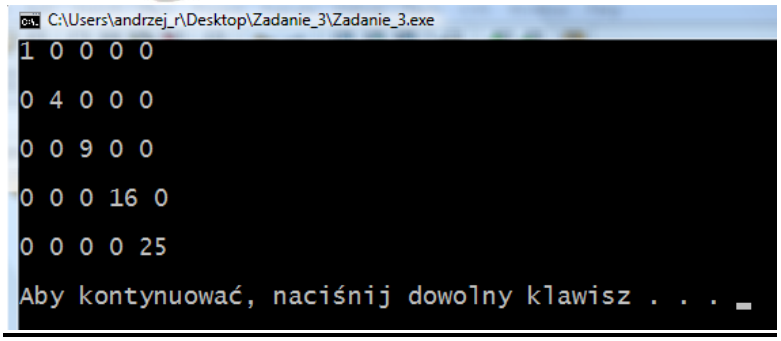


ZADANIE 3

Wygeneruj następującą tablicę używając pętli podwójnej.

1 0 0 0 0
0 4 0 0 0
0 0 9 0 0
0 0 0 16 0
0 0 0 0 25

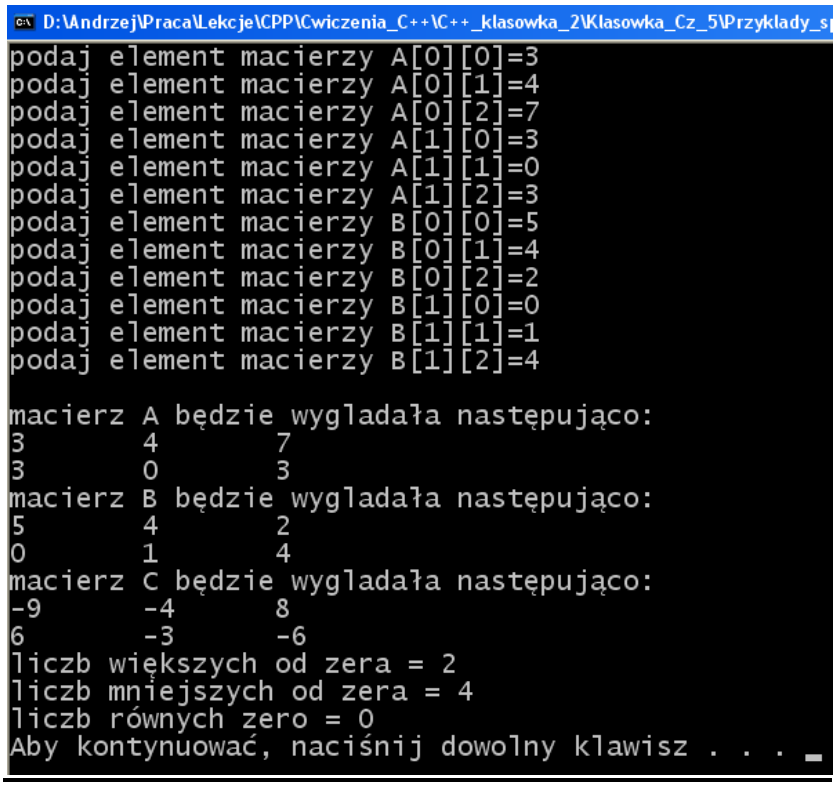
oraz wydrukować na ekranie wierszami.



ZADANIE 4

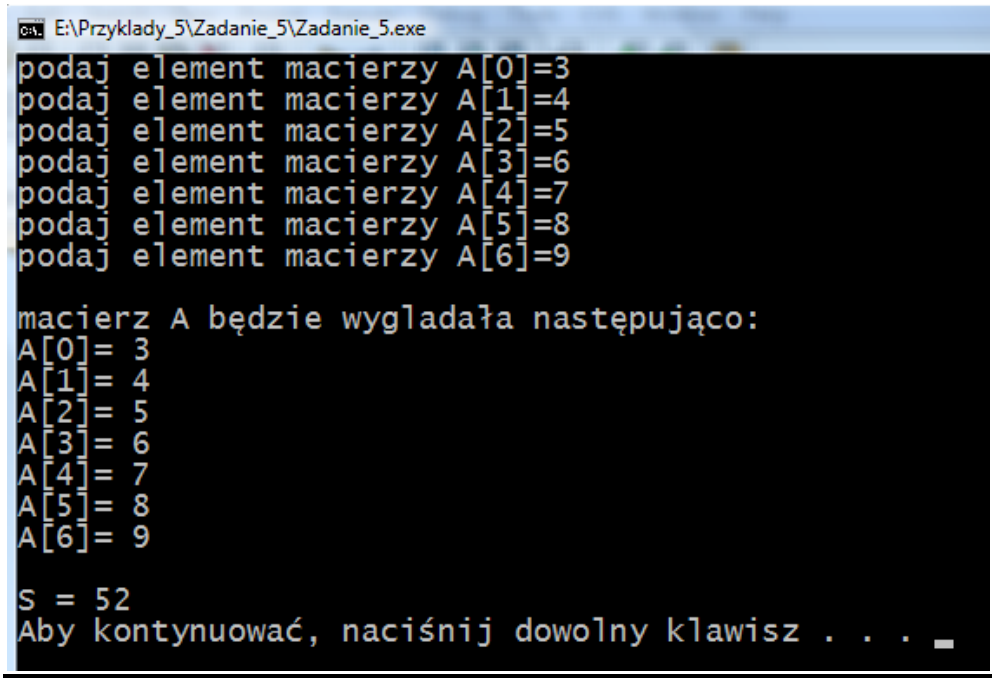
Wczytaj dwie macierze (tablice) A i B, obie o wymiarze 2*3.

- Wypisz wierszami macierze A B.
- Oblicz macierz C=2A–3B.
- Wypisz wierszami macierze C.
- Następnie program powinien zliczać ilość wyrazów macierzy(tablicy) C większych od zera, równych zero oraz mniejszych od zera.



ZADANIE 5

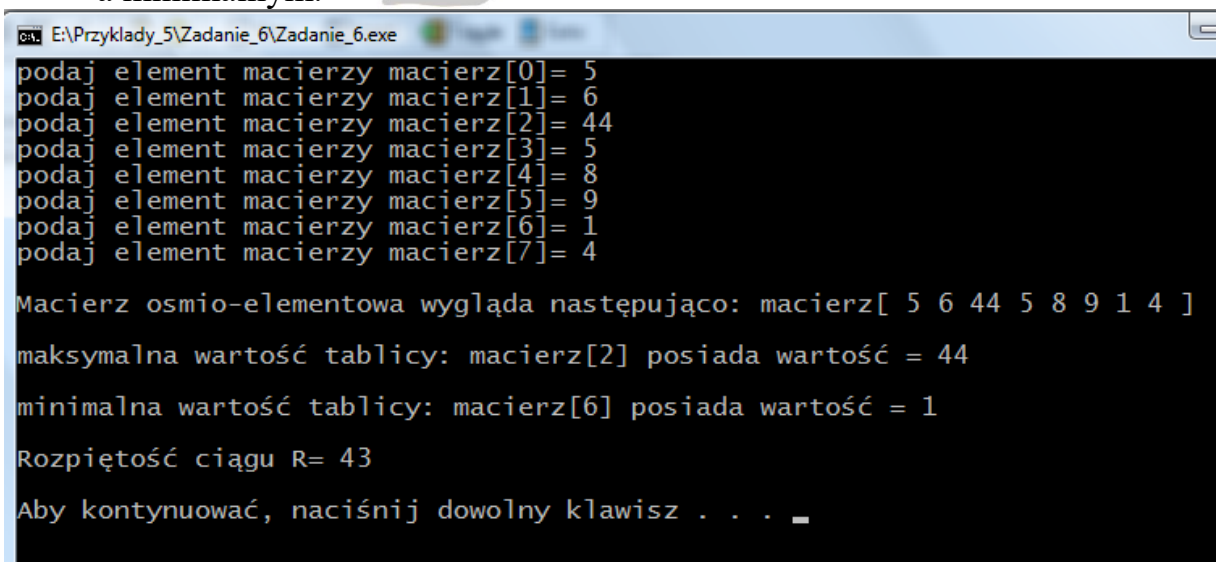
Wczytaj dowolne dane do tablicy jednowymiarowej $\mathbf{A_i}$ siedmio elementowej bez użycia pętli.		
Wypisz wczytaną tablicę w następujący sposób:	A[0]=2 A[1]=1 …… A[6]=5	Oblicz następującą sumę $S = \sum_{3}^{6} (2 * A_i - 2)$



ZADANIE 6

Oblicz rozpiętość ciągu ośmio-elemnetowego wczytanego z klawiatury do tablicy jednowymiarowej .

- Wypisz ten ciąg wyraz po wyrazie w tablicy jednowymiarowej.
- Wypisz który element ciągu jest największy a który najmniejszy, podczas wypisywania podaj indeks konkretnego elementu macierzy;
- Podaj rozpiętość ciągu: jest to różnica między elementem maksymalnym a minimalnym.



ZADANIE 7

Napisz program który sprawdzi czy podana liczba należy do ciągu Fibonacciego.

```
Będę sprawdzał czy podana liczba mależy do ciągu Fibonacciego.
Podaj liczbę do sprawdzenia:
15
podana liczba 15 nie należy dociągu Fibonacciego
Aby kontynuować, naciśnij dowolny klawisz . . .
```

```
Będę sprawdzał czy podana liczba mależy do ciągu Fibonacciego
Podaj liczbę do sprawdzenia:
8
6 wyraz ciągu Fibonacciego wynosi: 8
Aby kontynuować, naciśnij dowolny klawisz . . .
```