

PWSZ Elbląg
Instytut Informatyki Stosowanej im. Krzysztofa Brzeskiego

Podstawy programowania II – laboratorium

Studium Stacjonarne, sem. 2, 2021/2022

Sprawozdanie nr 2,
środa grupa II, 9:30 – 11:00

Data wykonania ćwiczenia: 19.03.2022

Data oddania sprawozdania: 19.03.2022

Nazwisko i imię: Gajewski Mikołaj

Nr albumu: 20180

Nazwa pliku : lab2_SD_Gajewski_Mikołaj_20180.pdf

1. a) Napisać funkcję wczytującą tablicę dwuwymiarową typu double o nagłówku void wczyt2D (int n, double x[][M} (M=2 - stała) z zabezpieczeniem formatu oraz funkcję drukującą tablicę dwuwymiarową void druk2D(int n, double x[][M} w postaci macierzowej.

b) Napisać funkcję wczytującą tablicę dwuwymiarową typu double w stylu C99 void wczyt2DC99 (int n, int m. double x[n][m]) z zabezpieczeniem formatu (n – liczba wierszy, m – liczba kolumn) oraz funkcję void druk2DC99 (int n, int m. double x[n][m]) drukującą tablicę dwuwymiarową w postaci macierzowej. W funkcji main() wywołać funkcje wczyt2D () i druk2D() dla tablicy z[2][2], a funkcje wczyt2DC99 i druk2DC99 dla tablic y[2][3] i z[2][4].

Rozwiązanie (a) :

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define M 2
```

```
/* run this program using the console pauser or add your own getch, system("pause") or input loop */
```

```
void wczyt2D(int n, double x[M][M]);
```

```
void druk2D(int n, double x[M][M]);
```

```
int main(int argc, char *argv[]) {
```

```
    double x[M][M];
```

```
    wczyt2D(M,x);
```

```
    druk2D(M,x);
```

```
    return 0;
```

```
}
```

```
void wczyt2D(int n, double x[M][M]){
```

```
    printf("\nPodaj wartosci do tablicy dwuwymiarowej : ");
```

```
    int i ,j;
```

```
    for(i =0; i < M; i ++)
```

```

        for( j=0; j<M; j++){
            while(scanf("%lf", &x[i][j])!= 1){
                printf("\n Bład formatu ! Proszę podać jeszcze raz :");
                fflush(stdin);
            }
            printf("Podaj kolejną wartość :");
        }
    }

void druk2D(int n, double x[M][M] ){
    int i,j ;
    printf("\nWynik w postaci macierzy : ");
    printf("\n");
    for ( i =0 ; i < M; i++){
        for (j =0; j < M; j++)
            printf("[%d][%d]=%lf", i, j, x[i][j]);
        printf("\n");
    }
    getchar();
}

```

Widok z terminala :

```

Podaj wartosci do tablicy dwuwymiarowej : 4
Podaj kolejna wartosc :4
Podaj kolejna wartosc :4
Podaj kolejna wartosc :3
Podaj kolejna wartosc :
Wynik w postaci macierzy :
[0][0]=4.000000[0][1]=4.000000
[1][0]=4.000000[1][1]=3.000000

-----
Process exited after 3.671 seconds with return value 0
Press any key to continue . . .

```

Rozwiązanie (b) :

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
/* run this program using the console pauser or add your own getch, system("pause") or input loop */
```

```
void druk2DC99 ( int n, int m, double x[n][m]);
```

```
void wczyt2DC99( int n, int m,double x[n][m]);
```

```
int main(int argc, char *argv[]) {
```

```
    printf("Wczytywanie do tablicy .");
```

```
    int x[2][2];
```

```
    wczyt2DC99(2,2,x);
```

```
    druk2DC99(2,2,x);
```

```
    return 0;
```

```
}
```

```
void wczyt2DC99( int n, int m,double x[n][m]){
```

```
    int i, j;
```

```
    printf("\nPodaj wartosc do tablicy dwuwymiarowej :");
```

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

```
        for( j=0; j<m; j++){
```

```
            while(scanf("%lf", &x[i][j])!= 1){
```

```
                printf("\n Blad formatu ! Prosze podac jeszcze raz :");
```

```

        fflush(stdin);
    }
    printf("\nPodaj kolejna wartosc :");
}
}

void druk2DC99 ( int n, int m, double x[n][m]){
    int i,j ;
    printf("\nWynik w postaci macierzy : ");
    printf("\n");
    for ( i =0 ; i < n; i++){
        for (j =0; j < m; j++){
            printf("[%d][%d]=%lf", i, j, x[i][j]);
            printf("\n");
        }
        getchar();
    }
}

```

Widok z terminala :

```

Wczytywanie do tablicy .
Podaj wartosc do tablicy dwuwymiarowej :5
Podaj kolejna wartosc :4
Podaj kolejna wartosc :3
Podaj kolejna wartosc :4
Podaj kolejna wartosc :
Wynik w postaci macierzy :
[0][0]=5.000000[0][1]=4.000000
[1][0]=3.000000[1][1]=4.000000
-----
Process exited after 6.793 seconds with return value 3221225477
Press any key to continue . . .

```

2. Napisać program inicjujący tablicę dwuwymiarową `tab (3 x 3) int tab[3][3]={ {1, 2, 3}, {2,5,7}, {0,1, -1}}`; oraz funkcję `void sumWKP(int n, int m, double x[n][m], int nrWiersza, int nrKolumny, double wyniki[])` obliczającą i zwracającą przy użyciu dodatkowej tablicy sumę elementów wybranego wiersza, sumę wybranej kolumny oraz sumę elementów na głównej przekątnej. W funkcji `main()` wywołać funkcję dla tablicy `tab`. Wywołanie ma mieć postać `sumWKP(3,3,tab, 1, 0, rezultaty)`; Numery wiersza i kolumny wczytać z klawiatury w funkcji `main()` z zabezpieczeniem formatu i zakresu dopuszczalnych wartości. Zdefiniować tablicę rezultaty jako `double rezultaty[3]`.
Wydrukować wyniki.

Rozwiązanie :

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
/* run this program using the console pauser or add your own getch, system("pause") or input loop */
```

```
void sumWKP(int n, int m, int x[n][m], int nrWiersza, int nrKolumny, double wyniki[]);
```

```
int main(int argc, char *argv[]) {
```

```
    int nrKolumny, nrWiersza;
```

```
    int i ;
```

```
    double rezultaty[3];
```

```
    int tab[3][3]={ {1, 2, 3}, {2,5,7}, {0,1, -1}};
```

```
    printf("\nPodaj numer wiersza z przedzialu 0 - > 2 : ");
```

```
    while(scanf("%d",&nrWiersza)!= 1 || nrWiersza < 0 || nrWiersza > 2){
```

```
        printf("Bledny format! Prosze podac jeszcze raz : ");
```

```
        fflush(stdin);
```

```
    }
```

```
    printf("\nWczytano : %d", nrWiersza);
```

```
    printf("\nPodaj numer Kolumny z przedzialu : 0->2 : ");
```

```
    while(scanf("%d",&nrKolumny) != 1 || nrKolumny < 0 || nrKolumny > 2){
```

```
        printf("Bledny format! Prosze podac jeszcze raz : ");
```

```
        fflush(stdin);
```

```
    }
```

```
    printf("\nWczytano : %d", nrKolumny);
```

```

sumWKP(3,3,tab,nrWiersza,nrKolumny, rezultaty );

for ( i= 0; i < 3 ; i++){
    printf("\ntab[%d]=%lf", i ,rezultaty[i]);
}

return 0;
}

void sumWKP(int n, int m, int x[n][m], int nrWiersza, int nrKolumny, double wyniki[]){
    int sumaKol = 0;
    int sumaW = 0;
    int j,i;
    int sumaPe=0;
    for ( i =0; i <n ; i++)
        sumaW+=x[nrWiersza][i];

    for(j =0;j<n;j++)
        sumaKol+=x[j][nrKolumny];

    for(i=0; i <n ; i++)
        sumaPe+=x[i][i];

    wyniki[0]=sumaKol;
    wyniki[1]=sumaW;
    wyniki[2]=sumaPe;

}

```

Widok z terminala :

```

Podaj numer wiersza z przedzialu 0 - > 2 : 2
Wczytano : 2
Podaj numer Kolumny z przedzialu : 0->2 : 1
Wczytano : 1
tab[0]=8.000000
tab[1]=0.000000
tab[2]=5.000000
-----
Process exited after 2.935 seconds with return value 0
Press any key to continue . . .

```

3. Napisać, wykorzystując funkcje z Zad.1

- funkcję zamienKK zamieniającą miejscami w tablicy dwie wybrane kolumny void zamienKK(int n, int m, double x[n][m], int nrKol1, int nrKol2);
- funkcję zamienWW zamieniającą miejscami w tablicy dwa wybrane wiersze void zamienWW(int n, int m, double x[n][m], int nrWiersza1, int nrWiersza2);
- funkcję transpose transponującą macierz (n x n) void transpose(int n, double x[n][n]);
- funkcję sarrus obliczającą wyznacznik macierzy 3x3 metodą Sarrusa. double sarrus(double x[][3]);

Rozwiązanie (a) :

```
#include <stdio.h>
```

```
void zamienKK( int n, int m , int x[n][m], int nrKo1, int nrKol2);
```

```
int main() {
```

```
    unsigned int i,j;
```

```
    int x[3][3]= { {5,6,1}, {0,4,8}, {-2,3,-1} };
```

```
    printf("\nWydruk tablicy przed zamiana kolumn :");
```

```
    printf("\n");
```

```
    for(i=0; i<3;i++){
```

```
        for (j=0;j<3;j++)
```

```
            printf ("%2d ", x[i][j]);
```

```
            printf("\n");
```

```
    }
```

```
    getchar();
```



```
zamienKK(3,3,x,0,1);
```

```
printf("\nWydruk tablicy po zamianie kolumn");
```

```
printf("\n");
```

```
    for(i=0; i<3;i++){
```

```
        for (j=0;j<3;j++)
```

```
            printf ("%2d ", x[i][j]);
```

```
            printf("\n");
```

```
    }
```

```
    getchar();
```

```
    return 0;
```

```
}
```

```
void zamienKK( int n, int m , int x[n][m], int nrKol1, int nrKol2){
```

```
    unsigned int i,j;
```

```
    int temp;
```

```
    for (i=0; i<3;i++){
```

```
        temp=x[i][nrKol1];
```

```
    x[i][nrKol1]=x[i][nrKol2];
```

```
    x[i][nrKol2]=temp;
```

```
    }
```

```
}
```

Widok z terminala :

```
Wydruk tablicy przed zamiana kolumn :
```

```
5 6 1  
0 4 8  
-2 3 -1
```

```
Wydruk tablicy po zamianie kolumn
```

```
6 5 1  
4 0 8  
3 -2 -1
```

Rozwiązanie (b) :

```
#include <stdio.h>

#define N 3

#define M 3

void zamienWW( int n, int m, int x[n][m], int nrWiersza1, int nrWiersza2);

int main() {

    int x[N][M]= {{5,6,1}, {0,4,8}, {-2,3,-1}};

    unsigned int i,j;

    printf("Wydruk tablicy przed zmiana wierszy : ");

    printf("\n");

    for(i=0; i<N;i++){

        for (j=0;j<M;j++)

            printf ("%2d ", x[i][j]);

        printf("\n");

    }

    zamienWW(N,M,x,0,1);

    printf("Wydruk tablicy po zamianie wierszy\n");

    printf("\n");

    for(i=0; i<N;i++){

        for (j=0;j<M;j++)

            printf ("%2d ", x[i][j]);

        printf("\n");

    }

    getchar();

    return 0;

}

void zamienWW( int n, int m, int x[n][m], int nrW1, int nrW2)

{

    unsigned int i,j;

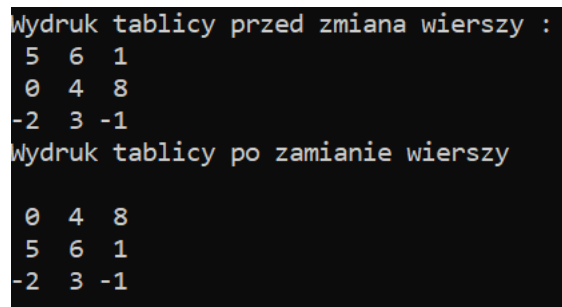
    int temp;
```

```

        for (j=0; j<N;j++)
    {
        temp=x[nrW1][j];
        x[nrW1][j]=x[nrW2][j];
        x[nrW2][j]=temp;
    }
}

```

Widok z terminala :



```

Wydruk tablicy przed zmiana wierszy :
5 6 1
0 4 8
-2 3 -1
Wydruk tablicy po zamianie wierszy
0 4 8
5 6 1
-2 3 -1

```

Rozwiązanie (c) :

```

#include <stdio.h>

#define N 3

void transpose( int n, int x[N][N]);

int main() {
    int x[N][N]= {{5,6,1}, {0,4,8}, {-2,3,-1}};
    unsigned int i,j;
    printf("Wydruk tablicy przed operacja (transpozycji) : ");
    printf("\n");
    for(i=0; i<N;i++){
        for (j=0;j<N;j++)
            printf ("%2d ", x[i][j]);
        printf("\n");
    }
    transpose(3,x);
    return 0;
}

```

```

void transpose(int n, int x[n][n]){
    int trans[3][3];
    unsigned int i, j;

    for (i = 0; i < 3; ++i){

        for (j = 0; j < 3; ++j) {
            trans[j][i] = x[i][j];
        }
    }

    printf("Wydruk tablicy po operacji \n");

    printf("\n");
    for(i=0; i<N;i++){
        for (j=0;j<N;j++)
            printf ("%2d ", trans[i][j]);
        printf("\n");
    }
    getchar();
}

```

Widok z terminala :

```

wydruk tablicy przed operacja (transpozycji) :
 5  6  1
 0  4  8
-2  3 -1
wydruk tablicy po operacji

 5  0 -2
 6  4  3
 1  8 -1

```

Rozwiązanie (d) :

```
#include <stdio.h>

#define N 3

double sarrus(int x[3][3]);

int main() {
    int x[N][N]= {{5,6,1}, {0,4,8}, {-2,3,-1}};
    unsigned int i,j;
    printf("Wydruk tablicy przed operacja (metoda Sarrusa) : ");
    printf("\n");
    for(i=0; i<N;i++){
        for (j=0;j<N;j++)
            printf ("%2d ", x[i][j]);
        printf("\n");
    }
    sarrus(x);
    return 0;
}

double sarrus(int x[N][N]){
    int det=0;
    int i, j;

    for(i=0;i<3;i++){
        det = det + (x[0][i]*(x[1][(i+1)%3]*x[2][(i+2)%3] - x[1][(i+2)%3]*x[2][(i+1)%3]));
    }

    printf("\nWyznacznik: %d", det);
}
```

Widok z terminala :

```

Wydruk tablicy przed operacja (metoda Sarrusa) :
 5  6  1
 0  4  8
-2  3 -1

Wyznacznik: -228
-----
Process exited after 0.01027 seconds with return value 0
Press any key to continue . . .

```

4. Zainicjować tablicę `x[4][4]` i napisać funkcję `void minmax2D (int n, double x[n][n],double wyniki[])`; przepisującą tablicę dwuwymiarową do tablicy jednowymiarowej, wyznaczającą i zwracającą wartość minimalną i wartość maksymalną. Funkcję wywołać w funkcji `main()`, wydrukować wyniki.

Rozwiązanie :

```
#include <stdio.h>
```

```
void minmax2D (int n, int x[n][n],int wyniki[ ]);
```

```
int main (){
```

```
    int x[4][4]={ { 1,321 },{ 3,0 },{ 7,4 },{ 4,10 } }; <- tablica 4x4
```

```
    int wyniki[8];
```

```
    minmax2D(4,x,wyniki);
```

```
}
```

```
void minmax2D (int n, int x[n][n],int wyniki[ ]){
```

```
    unsigned int j,i,k=0, n1=4;
```

```
    int amin;
```

```
    int c;
```

```
    int p;
```

```
    for (i=0;i<n1;i++)
```

```
        for (j=0;j<n1;j++)
```

```
            if(x[i][j]!=0)
```

```
                wyniki[k++]=x[i][j];
```

```
    printf(" \nTablica jednowymiarowa koncowa : ");
```

```

for (i=0;i<8; i++)

    printf(" %u ", wyniki[i]);

getchar();

for (i=0;i<8;i++){
    amin=wyniki[i];
    c=i;
    for (j=i+1;j<8;j++)
        if (wyniki[j]<amin){
            amin=wyniki[j];
            c=j; }
    if (c!=i){
        p=wyniki[i];
        wyniki[i]=wyniki[c];
        wyniki[c]=p;
    }

}

printf("\nStan tablicy po sortowaniu :");

for ( i =0; i < 8 ; i++){
    printf("\ntab[%d]=%d", i, wyniki[i]);
}

printf("\nwartosc maksymalna : %d", wyniki[7]);
printf("\nwartosc minimalna : %d", wyniki[0] );

}

```

Widok z terminala :

```
Tablica jednowymiarowa koncowa : 1 321 3 7 4 4 10 0
```

```
Stan tablicy po sortowaniu :
```

```
tab[0]=0
```

```
tab[1]=1
```

```
tab[2]=3
```

```
tab[3]=4
```

```
tab[4]=4
```

```
tab[5]=7
```

```
tab[6]=10
```

```
tab[7]=321
```

```
wartosc maksymalna : 321
```

```
wartosc minimalna : 0
```

```
-----
```

```
Process exited after 1.469 seconds with return value 22
```

```
Press any key to continue . . .
```