Dynamiczna alokacja tablic w C

Infromatyka 1 Piotr Darnowski

21.12.2015

W ćwiczeniu 7 z C wykonujemy dynamiczną alokację tablic – istnieje kilka możliwości wykonania tego zadania. Poniżej przedstawiono trzy metody. Pierwsza z zastosowaniem dwóch tablic jednowymiarowych, druga z zastosowaniem jednej tablicy dwuwymiarowej i trzeci z zastosowaniem tablicy dwuwymiarowej i operatora new (który jednak jest częścią C++).

Plik1.txt to plik z ćwiczeń.

Treść programu:

```
#include <stdio.h>
#include <math.h>
#include <time.h>
#include <stdlib.h>
#include "winbgi2.h"
//#define _CRT_SECURE_NO_DEPRECATE //Dla MS Visual 2013 jeżeli są problemy z plikami


void main()
{
 /* Wczytujemy plik plik1 i sprawdzamy czy nie jest pusty */
         FILE *plik = fopen("plik1.txt","r");
         int N=0, Nall=0;      //deklaracja rozmiarow pliku

        if(plik == NULL)
        {
                printf("Blad otwarcia pliku\n");      exit(-1);
        }

/* Wczytujemy z klawiatury ile wierszy z pliku wczytamy  */
        printf("Podaje ile wierszy z pliku plik1.txt chcesz wczytać (nie wiecej niz 255) \n");
        scanf("%d",&N);

/*****************************************************************************/
/* Podejscie #1 - dynamiczna alokacja dwoch tablic jednowymiarowych */
        printf("Podejscie #1\n");
        double *x1 = (double*)malloc(N*sizeof(double)); //Przydzielamy dwa razy pamiec
        double *y1 = (double*)malloc(N*sizeof(double));

        fscanf(plik,"%d",&Nall);
        printf("Plik zawiera %d wierszy danych \n", Nall);

        /* Wczytujemy i wyswietlamy plik */
        for (int i=0;i<N;i++)
        {
         fscanf(plik,"%lf      %lf",&x1[i],&y1[i]);
         printf("wiersz %d:  %lf %lf\n",i+1, x1[i],y1[i]); //i+1 dla numeracji
        }

        free(x1); free(y1); //Zwalniamy pamiec
        fclose(plik);          //zamykamy plik
        getchar();
/*****************************************************************************/
/* Podejscie #2 - dynamiczna alokacja jednej tablicy dwuwymiarowej */
        printf("Podejscie #2\n");
        FILE *plik1 = fopen("plik1.txt","r");         //Zamknelismy plik otwieramy od nowa
        double **xy;                          //Deklaracja wskaznika do tablicy dwuwymiarowej
        //przydzielamy miejsce w pamieci (rozmiar N dalej obowiazuje)

        xy = (double**)malloc(N*sizeof(double));

        /*Zamknelismy plik wiec czytamy go od nowa. Wczesniej wczytywalismy linia po lini.
        Teraz aby nie wczytywac ponownie N omijamy pierwszą linie.
```

```
        Sluzy do tego funkcja fseek*/

        //Operujemy na pliku: plik, czwarta pozycja (drugi wiersz zaczyna się od 4 pozycji),
        od początku pliku (0).

        fseek(plik1,4,0);



        /* Dynamicznie alokujemy pamiec na dwie zmienne typu double na kazdy wiersz tablicy*/
        for (int i=0;i<N;i++)
        {
        //dynamicznie alokujemy pamiec na dwie (!) zmienne typu double
               xy[i] = (double*)malloc(2 * sizeof(double));
        }

        /* Czytamy i wyswietlamy plik do tablicy dwuwymiarowej i w
        ten sposób calosc wczytujemy do jednej tablicy 2D a nie dwoch 1D jak wczesniej*/
        for (int i=0;i<N;i++)
        {
               fscanf(plik1,"%lf      %lf",&xy[i][0],&xy[i][1]);
               printf("Wiersz: %d     %lf     %lf\n", i+1, xy[i][0],xy[i][1]);
        }
        //Zwalniamy pamiec
        for(int i=0;i<N;++i) free(xy[i]);
                free(xy);

        //fclose(plik);//specjalnie NIE zamykamy pliku - dla przypadku#3 wykorzystamy fseek
        getchar();

/*****************************************************************************/
/* Podejscie #3 - dynamiczna alokacja jednej tablicy dwuwymiarowej inny
  podobny sposób z zastosowaniem operatorów new - któe de facto są standardem C++ ale ich
użycie wydaje się bardziej przyswajalne*/

        printf("Podejscie #3\n");
        fseek(plik1,4,0);      //Wracamy do 4 pozycji w pliku
        int N2 = 2;                    //"Drugi wymiar tablicy"

        double **xy1;  //Pododbnie jak poprzednio deklarujemy wskaznik

        //Zamiast malloc stosujemy operator new
        xy1 = new double*[N]; //N dla pierwszego wymiaru tablicy

        for(int i=0;i<N;i++)
        {
         xy1[i] = new double[N2]; //Drugi wymiar tablicy - "rozciagamy" ja na drugi wymiar
        }

        for (int i=0;i<N;i++)
        {
         fscanf(plik1,"%lf     %lf",&xy1[i][0],&xy1[i][1]);
         printf("Wiersz: %d    %lf     %lf\n", i+1, xy1[i][0],xy1[i][1]);
        }

        //Zwalniamy pamiec
        for(int i=0;i<N;++i) free(xy1[i]);
                free(xy1);
        fclose(plik1); //Zamykamy plik
        getchar();
/*****************************************************************************/
//Analogicznie mozna deklarowac tablice 3 i wiecej wymiarowe



        getchar();
} //KONIEC MAIN
```