Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра информационных технологий автоматизированных систем

Факультет ИТиУ  
Cпециальность АСОИ

Индивидуальная практическая работа по модулю 1

по дисциплине «Системное программное обеспечение», часть 1

«Управление процессами и потоками»

Вариант №1

Выполнил:

Ст. Гр. 820601

Шведов А.Р

Зачетная книжка No 82060145

Минск 2020

# Задание

Во всех вариантах необходимо написать три функции, которые будут вызываться из функции main(). Предусмотреть обработку исключительных ситуаций на этапах ввода и вывода данных и во время вычислений

**Первая** функция получает размерность массива, создает динамический массив и возвращает указатель на начало созданного массива.

**Вторая**– получает адрес массива и его размерность и решает одну из ниже перечисленных задач.

**Третья** функция получает адрес массива и его размерность и освобождает память, занятую массивом.

Размерность вводится с клавиатуры в функции main() и передается в первую функцию. Значения элементов вводятся с клавиатуры в первой функции.

В заданиях с нечетным номером использовать функции управления памятью библиотеки языка С (#include ). В заданиях с четным номером использовать функции управления памятью Win32 API (**#include )**.   
  
Задача для решения:

Массив размерностью MxN. Необходимо найти наибольший и наименьший элементы.

# Ход работы

Листинг программы:

**Файл “functions.hpp”**

#ifndef functions\_hpp  
#define functions\_hpp  
  
struct Size {  
 int M;  
 int N;  
};  
int\* createArray(Size size);  
void find\_min\_max(int \*array, Size size);  
void removeArray(int \*array, Size size);

int enter();  
#endif /\* functions\_hpp \*/

**Файл “functions.cpp”**

#include "functions.hpp"  
#include <iostream>  
#include <stdio.h>  
#include <stdlib.h>  
  
using namespace std;  
  
int\* createArray(Size size) {  
 int \*array = (int \*)malloc(size.M \* size.N \* sizeof(int));  
  
 cout << "enter array values:" << endl;  
 for (int i = 0; i < size.M; i++) {  
 for (int j = 0; j < size.N; j++) {  
 cout << "arr[" << i << "][" << j << "]: ";  
 cin >> array[i \* size.N + j];  
 cout << endl;  
 }  
 }  
   
 return array;  
}  
  
void find\_min\_max(int \*array, Size size) {  
 int min = array[0];  
 int max = array[0];  
   
 for (int i = 0; i < size.M; i++) {  
 for (int j = 0; j < size.N; j++) {  
 if (array[i \* size.N + j] < min) {  
 min = array[i \* size.N + j];  
 }  
   
 if (array[i \* size.N + j] > max) {  
 max = array[i \* size.N + j];  
 }  
 }  
 }  
   
 cout << "max: " << max << endl;  
 cout << "min: " << min << endl;  
}

void removeArray(int \*array, Size size) {  
 free(array);  
}

int enter(){ // catch int input errors  
 int n;  
  
 while(true){  
 cin >> n;  
 if (!cin){ // == cin.fail()  
 cout << "error. Try again\n";  
 cin.clear();  
 while(cin.get()!='\n');  
 }else break;  
 }  
 return n;  
}

**Файл “main.cpp”**

#include <iostream>  
#include "functions.hpp"  
  
using namespace std;  
  
int main(int argc, const char \* argv[]) {  
 Size size{};  
   
 try {  
 cout << "Enter the size of an array MxN " << endl;  
 cout << "M: ";  
 size.M = enter();  
  
 if (size.M <= 0) {  
 throw "M should be > 0";  
 }  
   
 cout << endl << "N: ";  
 size.N = enter();  
   
 if (size.N <= 0) {  
 throw "N should be > 0";  
 }  
 cout << endl;  
   
 } catch(const char\* msg) { // catch errors  
 cerr << msg << endl;  
 return 1;  
 }  
   
 int \*array = createArray(size);  
 find\_min\_max(array, size);  
 removeArray(array, size);  
   
 return 0;  
}

# Выводы

В ходе выполнения данной работы нами были закреплены работа по динамическому выделению памяти и обработке исключительных ситуаций в языке C++. Была написана программа по созданию массива, работе с ним с обработкой ошибок ввода данных.