Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра прикладной математики и кибернетики

ОТЧЕТ
по лабораторной работе 1

по дисциплине «**Объектно-ориентированное программирование**»

Выполнил:
студент гр. ИС-242
«11» сентября 2023 г. _____ /Любицкий М.Е./

Проверил:
Доцент Кафедры ПМиК
«11» сентября 2023 г. _____ /Ситняковская Е.И./

Оценка «_____»

Новосибирск 2023

ОГЛАВЛЕНИЕ

ЗАДАНИЕ

Задание 1.

Создать одномерный динамический массив типа int, заполнить его случайными числами, вывести на экран. Размер массива необходимо хранить в первом элементе массива.

Необходимо реализовать следующие функции:

```
int* genRandArray(int size, int maxValue);
void print(int* arr);
```

main должен выглядеть следующим образом:

```
int main(){
    int size = rand()%10;
    int maxValue = 100;
    int* arr = genRandArray(size, maxValue);
    print(arr);
    //очистка выделенной памяти
}
```

Пример вывода:

7: 44 11 24 41 10 57 100

Задание 2.

Создать двумерный динамический массив типа int, заполнить его случайными числами, вывести на экран. Размер каждого одномерного массива – произвольный (матрица не обязана быть прямоугольной). Количество строк матрицы хранить в отдельной переменной в main.

Функции, реализованные в первом задании, рекомендуется использовать во втором.

Необходимо реализовать следующие функции:

```
int** genRandMatrix(intsize, intmaxValue);
void printMatrix(int** matrix);
```

```
int main(){
    int size=rand()%10;
    int maxValue = 100;
    int** matrix = genRandMatrix(size, maxValue);
```

```
        print(matrix);
        //очистка памяти
}
```

Пример вывода:

```
 4
1: 15
5: 54 23 15 5 12
7: 1 32 51 42 51 100 12
3: 50 37 17
```

ВЫПОЛНЕНИЕ РАБОТЫ

Задание 1:

Функция `int * genRandArray(int size, int maxValue)`

Функция принимает на вход размер массива и максимальное значение в массиве.
В теле функции выделяем память под элементы массива и еще один под размер массива, который нужно хранить в первом элементе массива. Если память выделилась, то возвращается указатель на сгенерированный массив, иначе возвращается нулевой указатель.

```
4   int * genRandArray(int size, int maxValue)
5   {
6       int * arr = new int[size+1];
7
8       if(!arr)
9       {
10          return nullptr;
11      }
12
13      arr[0] = size;
14      for (int i = 1; i < size + 1; i++)
15      {
16          arr[i] = rand() % maxValue;
17      }
18
19      return arr;
20  }
```

Функция `void print(int * arr)`

Функция принимает на вход указатель на массив и выводит его размер и элементы массива.

```
39  void print(int * arr)
40  {
41      int size = arr[0];
42      std::cout << size << ": ";
43      for (int i = 1; i < size + 1; i++)
44      {
45          std::cout << arr[i] << ' ';
46      }
47      std::cout << '\n';
48  }
```

В теле функции main вызваем две предыдущие функции и после освобождаем выделенную память на наш массив.

```
63
64      int size = rand()%10;
65      int maxValue = 100;
66      int* arr = genRandArray(size, maxValue);
67      print(arr);
68      delete [] arr;
```

После запуска видим такой результат:

```
root@DESKTOP-BK46MSC:/mnt/d/Files/prog/oop_lab1# ./app
4: 82 38 58 64
```

Задание 2:

Функция `int ** genRandMatrix(int size, int maxValue)`

Функция принимает на вход количество строк в матрице и максимальное значение, которое будет в этой матрице. В теле функции выделяем массив указателей, после используем функцию из предыдущего задания и генерируем по массиву по адресу каждого указателя. Если память выделилась, то возвращаем указатель на матрицу, иначе нулевой указатель.

```
22  int ** genRandMatrix(int size, int maxValue)
23  {
24      int ** arr = new int*[size];
25
26      if(!arr)
27      {
28          return nullptr;
29      }
30
31      for (int i = 0; i < size; i++)
32      {
33          arr[i] = genRandArray(rand()%10, maxValue);
34      }
35
36      return arr;
37  }
```

Функция `void printMatrix(int ** arr, int size)`

Функция принимает указатель на матрицу и количество строк в этой матрице. Далее в теле цикле вызываем функцию из предыдущего задания для вывода массива. По итогу получаем вывод всей матрицы.

```
50  void printMatrix(int ** arr, int size)
51  {
52      std::cout << "size matrix: " << size << '\n';
53      for(int i = 0; i < size; i++)
54      {
55          print(arr[i]);
56      }
57      std::cout << '\n';
58  }
```

В теле функции main вызваем две предыдущие функции и после освобождаем выделенную память на нашу матрицу.

```
72      size=rand()%10;
73      int** matrix = genRandMatrix(size, maxValue);
74      printMatrix(matrix, size);
75      for (int i = 0; i < size; i++)
76      {
77          delete [] matrix[i];
78      }
79      delete [] matrix;
```

После запуска программы видим такой результат:

```
size matrix: 6
9: 14 25 15 52 51 52 98 63 78
0:
6: 19 40 61 90 33 30
3: 34 23 56
8: 69 8 12 51 98 23 67 34
2: 81 60
```

ПРИЛОЖЕНИЕ

Исходный код с комментариями:

main.cpp

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

int * genRandArray(int size, int maxValue)
{

    if(!arr)
    {
        return nullptr;
    }

    int * arr = new int[size+1];
    arr[0] = size;
    for (int i = 1; i < size + 1; i++)
    {
        arr[i] = rand() % maxValue;
    }

    return arr;
}

int ** genRandMatrix(int size, int maxValue)
{
    int ** arr = new int*[size];

    if(!arr)
    {
        return nullptr;
    }

    for (int i = 0; i < size; i++)
    {
        arr[i] = genRandArray(rand()%10, maxValue);
    }

    return arr;
}

void print(int * arr)
{
    int size = arr[0];
    std::cout << size << ": ";
    for (int i = 1; i < size + 1; i++)
    {
        std::cout << arr[i] << ' ';
    }
    std::cout << '\n';
}
```

```
49
50  void printMatrix(int ** arr, int size)
51  {
52      std::cout << "size matrix: " << size << '\n';
53      for(int i = 0; i < size; i++)
54      {
55          print(arr[i]);
56      }
57      std::cout << '\n';
58  }
59
60  int main()
61  {
62      srand(time(NULL));
63
64      int size = rand()%10;
65      int maxValue = 100;
66      int* arr = genRandArray(size, maxValue);
67      print(arr);
68      delete [] arr;
69
70      std::cout << "---------------\n";
71
72      size=rand()%10;
73      int** matrix = genRandMatrix(size, maxValue);
74      printMatrix(matrix, size);
75      for (int i = 0; i < size; i++)
76      {
77          delete [] matrix[i];
78      }
79      delete [] matrix;
80
81      return 0;
82  }
```