Санкт-Петербургский политехнический университет Петра Великого
Физико-механический институт
Высшая школа фундаментальных физических исследований

Лабораторная работа

Класс динамического массива на C++

По дисциплине ”Специальный практикум”

Выполнил
студент гр. 5040302/10301 С. А. Буланова

Научный руководитель:
д.ф.-м.н. Я. А. Бердников

«____» _____ 2021 г.

Санкт-Петербург
2021

Задание

Создание класса динамического массива

1. Элементы должны иметь тип DynamicArrayValue_t
2. У класса должны быть конструкторы:
   (a) Конструктор по умолчанию (в массиве 0 элементов, память не выделяется)
   (b) Конструктор, который создает массив из n элементов и заполняет их значениями val
   (c) Конструктор копирования
   (d) Оператор присваивания.
3. У класса должен быть деструктор
4. У класса должны быть методы:
   (a) Append(DynamicArrayValue_t val) – добавляет значение val в конец массива
   (b) Resize(size_t newSize) – увеличивает максимальное количество элементов
   (c) Get(size_t idx) – возвращает значение элемента

Приведем полученный код:

```
1  //DynamicArray.cpp
2
3  #include <iostream>
4  #include <cstdlib>
5  using namespace std;
6  using DynamicArrayValueType_t = int;
7
8  class DynamicArray{
9  private:
10   size_t fSize;
11   DynamicArrayValueType_t * fData;
12 public:
13   DynamicArray(){
14     fSize = 0;
15     fData = nullptr;
16     cout << "Default Constructor worked" << endl;
17   } //default constructor
18
19   DynamicArray(size_t n, const DynamicArrayValueType_t & val){
20     fSize = n,
21     fData = new DynamicArrayValueType_t[fSize];
22     for (size_t idx = 0; idx < fSize; idx++){
23       fData[idx] = val;
24     }
25     cout << "Array from " << n << " elements created" << endl;
26   } //constructor with values
27
28   DynamicArray(const DynamicArray & dyn_arr){
29     fSize = dyn_arr.fSize,
30     fData = new DynamicArrayValueType_t[fSize];
31     for (size_t idx = 0; idx < fSize; idx++){
32       fData[idx] = dyn_arr.fData[idx];
33     }
34     cout << "Copy constructor copied your object" << endl;
35   } //copy constructor
36
37   DynamicArray & operator = (const DynamicArray & dyn_arr){
38     fSize = dyn_arr.fSize;
```

```
39      fData = new DynamicArrayValueType_t[fSize];
40      for (size_t idx = 0; idx < fSize; idx++){
41        fData[idx] = dyn_arr.fData[idx];
42      }
43      cout << "Overload assignment operator created object" << endl;
44      return * this;
45    } //overload assignment operator
46
47    size_t Size() const {return fSize;}
48    DynamicArrayValueType_t Get(size_t idx) const {return fData[idx];}
49
50    ~DynamicArray(){
51      if (fData){
52        delete [] fData;
53        //cout << "Array deleted" << endl;
54      }
55    }
56
57     void Resize(size_t NewSize){
58       size_t n = NewSize;
59       if (fSize < NewSize){
60         for (size_t idx = fSize; idx < NewSize; idx++){
61           fData[idx] = 0;
62         }
63         for (size_t idx = 0; idx < NewSize; idx++){
64           cout << fData[idx] << ", ";
65         }
66         cout << endl;
67       }
68     }
69
70     void Append(const DynamicArrayValueType_t & val){
71       Resize(fSize++);
72       fData[fSize - 1] = val;
73     }
74
75     void Print() const {
76       for (size_t idx = 0; idx < fSize; idx++){
77         cout << fData[idx] << ", ";
78       }
```

```
79      cout << endl;
80    }
81
82 };
83
84 int main(){
85   DynamicArray a1;
86   a1.Print();
87   cout << "================" << endl;
88   DynamicArray a2(10, 60);
89   a2.Print();
90   cout << "================" << endl;
91   DynamicArray a3(a2);
92   a3.Print();
93   cout << "================" << endl;
94   DynamicArray a4(10, 9);
95   DynamicArray a5;
96   a5 = a4;
97   cout << "================" << endl;
98   a5.Resize(20);
99   a5.Append(999);
100  a5.Print();
101  cout << "Element #10 is: " << a5.Get(10) << endl;
102  return 0;
103 }
```

Продемонстрируем работу программы: