|  |
| --- |
| **Assignment # 8**  ***Session*: Spring 2022 *Total marks*: 100**  ***Name*** : ***\_\_nimra maqbool\_\_ Roll number* : bsce21012\_\_\_** |

***Submission:***

• *Email instructor or TA if there are any questions. You cannot look at others’ solutions or use others’ solutions, however, you can discuss it with each other. Plagiarism will be dealt with according to the course policy.*

*• Submission after due time will not be accepted.*

**There should be a Report explaining your code and highlighting results. Follow this naming convention for your report RollNumber\_Assignment#.pdf e.g BSCE21001\_Assignment8.pdf.**

**TASKS:**

Implement 'sortedArray' class. [20 Marks]

This class will have two data members: array and capacity

Take size of array to be 10

'capacity' represents how many elements are present in the array, its initial value will be zero.

This class will have the the following functions:

1- Constructor to initialize the array elements with -1 and capacity to zero [2]

2- The user will be able to insert a number until maximum size is reached and

Every value should be inserted in a way that does not disturb the sorted order [5]

i.e. void insertElement(int value)

3- Check whether the capacity has been approached. User will not be allowed to insert [1.5+1]

element if the array is already full

i.e. bool full()

4- The user will be able to print the entire array to check whether the order is correct or not [1]

i.e. void print()

5- The user will be able to search the inserted number [2.5]

i.e. int search(int value)

6- The user will be able to delete an element from the array taking index as parameter [3.5]

i.e. void deleteElementByIndex(int index)

Elements on the right of deleted element should be shifted towards left after the deletion

7- Check whether the array is empty. User will not be allowed to delete element if the array [1.5+1]

is already empty

i.e. bool empty()

8- Destructor [1]

|  |
| --- |
| **Function.h:**  #include <iostream>  using namespace std;  class sortedArray { private:  int capacity; //declaring  int array[10]; public:  sortedArray(int c, int ptr[10]) {  c = 0; //initializing capacity at zero  capacity = c;  for (int i = 0; i < 10; i++) { //setting values of array at -1  ptr[i] = -1;  array[i] = ptr[i];  }  cout << "PARAMETRIZED CONSTRUCTOR IS CALLED.." << endl;  }   void insertElement(int value) {  int j; //declaring  cout << "PLEASE ENTER ELEMENTS OF ARRAY = ";  for (int i = 0; i < 10; i++) {  cin >> array[i]; //taking input in array and incrementing capacity after every input  capacity++;  if(capacity==10){  cout<<"YOU CAN'T ENTER MORE ELEMENTS IN ARRAY.."<<endl;  Break;  }  }  for (int i = 0; i < 10; i++) {  value = array[i]; //setting value at zero index of array  for (j = i - 1; j >= 0 && array[j] > value; j--) { //checking that if the next value is greater  array[j + 1] = array[j]; //copying  }  array[j + 1] = value; //copying  }  }   bool full() {  if (capacity == 10) { //checking that if the capacity is full then it can't enter more inputs  cout << "THE CAPACITY IS EQUAL TO THE SIZE" << endl;  cout << "YOU CAN'T ENTER MORE ELEMENTS.." << endl;  return true;  } else { //else you can  cout << "THERE IS SPACE IN THE ARRAY YOU CAN ENTER MORE ELEMENTS" << endl;  return false;  }  }   void print() {  for (int i = 0; i < 10; i++) {  cout << array[i]<<" "; //displaying by loop   }  cout << "\ncapacity =" << capacity << endl; //displaying capacity  }   void search(int value) {  cout<<"ENTER ELEMENTS OF ARRAY = ";  for (int i = 1; i <=10; i++) { //taking input in loop and incrementing capacity  cin >> array[i];  capacity++;  }  cout << "ENTER THE NUMBER YOU WANT TO SEARCH = ";  cin >> value; //taking input the value user want to search  for (int i = 1; i <=10; i++) { //applying loop  if (array[i]==value) { //checking that if array of i is equal to the value  cout << "THE ELEMENT IS PRESENT IN ARRAY AT INDEX = " << i << endl; //then display index at which the value is present  break;  }  if(i==capacity){ //if i is equal to capacity  cout<<"THIS NUMBER IS NOT PRESENT IN ARRAY.."<<endl;  break;  }  }   }  void deleteElement(int index){  cout<<"ENTER ELEMENTS = ";  for(int i=0;i<10;i++){  cin>>array[i]; //taking input  }  cout<<"ENTER INDEX = ";  cin>>index; //taking index which user wants to delete  for (int i = index; i <9 ; i++) {  if(array==0){ //A check to check that if the array is empty  cout<<"THE ARRAY IS EMPTY YOU CAN'T DELETE ANY ELEMENT.."<<endl;  }  else{  array[i] = array[i + 1]; //else move the element to left index  }  }  for(int i=0;i<10;i++){  cout<<array[i]<<" "; //displaying  }  cout<<"\nIT IS GIVING GARBAGE VALUE AT LAST INDEX"<<"\nAS THE ELEMENTS ARE MOVED TO LEFT AFTER DELETION.."<<endl;  }  bool empty(){  if(array==0){  cout<<"THE ARRAY IS EMPTY.."<<endl; //checking that if the array is empty or not  return true;  }  else{  cout<<"THERE ARE ELEMENTS IN THE ARRAY .."<<endl; //else return false  return false;  }  }  ~sortedArray(){  cout<<"DESTRUCTOR CALLED.."<<endl;  }  };   #endif //INC\_2022\_SPRING\_CE\_OOP\_WEEK8\_ASSIGNMENT8\_BSCE21012\_FUNCTIONS\_H   * In function.h I have made a class of sorted array havimg capacity and array as private members * To access those members, I have made a constructor in which set the capacity to zero and all elements of array to -1. * After that I have made an funtion of insert elements having a value in its prototype, after that I applied and loop in which taken the input and incremented the capacitance and checking that if capacitance is equal to the 10 then break * After that I have equaled the value to 1 index of array and then check that if next value is greater than the value then inserts that value to j+1 that is 1. * Then copied the value to the array of I index. * In the full function I have checked the capacity of the array if it is full then do not enter more elements and return true otherwise false * In print function I have printed array. * In the searching function I have enter elements in array then ask user to enter number and then checking each index of the array (if equal), then return the index. * In deleting function, I have deleted the index entered by the user and then moved the i+1 index to i index. * Then I have checked that if the array is empty then we can’t delete array. * Then I have made destructor.   **Main.cpp:**  #include <iostream> #include "Functions.h"  using namespace std;  int main() {  int c;  int ptr[10];  int value;  int opt;  sortedArray s(c, ptr);  do{  cout<<endl;  cout<<"WHICH TASK DO YOU WANT TO PERFORM?"<<endl;  cout<<"1.CONSTRUCTOR.."<<endl;  cout<<"2.SORT THE ARRAY AND PRINT IT.."<<endl;  cout<<"3.CHECK THAT ARRAY IS FULL OR NOT.."<<endl;  cout<<"4.PRINT THE ARRAY.."<<endl;  cout<<"5.SEARCH AN ELEMENTS FROM THE ARRAY.."<<endl;  cout<<"6.DELETE AN ELEMENT FROM AN SPECIFIC INDEX.."<<endl;  cout<<"7.CHECK THAT ARRAY IS EMPTY OR NOT.."<<endl;  cout<<"8.DESTRUCTOR.."<<endl;  cout<<"9.EXIT.."<<endl;  cin>>opt;  switch(opt){  case 1:{  sortedArray s(c, ptr);  break;  }  case 2:{  s.insertElement(value);  s.print();  break;  }  case 3:{  s.full();  break;  }  case 4:{  s.print();  break;  }  case 5:{  s.search(value);  break;  }  case 6:{  int index;  s.deleteElement(index);  break;  }  case 7:{  s.empty();  break;  }  case 8:{  s.~sortedArray();  break;  }  case 9:{  cout<<"YOU CHOOSE TO EXIT..."<<endl;  exit(2);  break;  }  default:{  cout<<"YOU HAVE ENTERED AN INVALID INPUT .."<<endl;  break;  }  }   }while(opt>=1 && opt<=9);    return 0; }   * In main I have made a menu and then called the function in the menu using switch cases.   **Output:**  **Text  Description automatically generated**  **Text  Description automatically generated**   * **I have put an break in the function due to which the menu breaks.**   **Text  Description automatically generated**  **Text  Description automatically generated** |