Name: Patel Manan Maheshkumar
Roll No: CE-111
PPS-II(Lab-7)

1. Which of the following can have a return type?
   (a) constructor (b) destructor (c) copy constructor (d) none of the above

   - (d) none of the above

2. State true/false with reason(s).

   (i) The only way an object can be passed to a constructor of the same class is a copy constructor.
   (ii) Constructors and destructors for the objects of a program are executed in the same order.

   - False, We can also pass an object by reference.
   - False, Constructors are executed in the order of object creation and Destructors are executed in the reverse order

3. Identify and correct the syntax and logical error(s) in class definition only in the following code to get output as 8.

```
#include <iostream>
using namespace std;
class Student{
public:
intRollNumber;
void PrintDetails(){
cout<<RollNumber<< "\n";
}
};
int main()
{
Student Student1;
Student1.RollNumber = 8;
const Student OtherStudent = Student1;
OtherStudent.PrintDetails();
}
```

- ```
  #include <iostream>
      using namespace std;

      class Student {
      public:
        int RollNumber;

        void PrintDetails() const {
          cout << RollNumber << "\n";
        }
      };

      int main() {
        Student Student1;
        Student1.RollNumber = 8;

        const Student OtherStudent = Student1;
        OtherStudent.PrintDetails();
    }
  ```

4. (i) Predict the output of the following program.
   (ii) What is missing in the program for its appropriate behaviour?
   (iii) Insert the appropriate feature in the program for the desired behaviour and show the change in the predicted output.
   (iv)How many times will the destructor of the class array be executed? Write a suitable destructor for the class array. Why is a user defined destructor desirable for this class?

```
#include <iostream>
using namespace std;
class list {
        int *t;
public:
        list() {
        t = new int[10];
for( int i = 0; i< 10; i++)
        t[i] = i;

}
void put(int i, int j) {
if( i>= 0 &&i< 10 )
        t[i] = j;
}
int get(int i) {
```
```

```
        return t[i];
    }
    };
    int main() {
    list beta;
    int i;
    for(i=9; i>=0; i--)
        cout<<beta.get(i) << "\n";
    list alpha(beta);
    for(i = 0; i< 10; i++)
        beta.put(i, i+1);
    for(i=0; i<10; i++)
        cout<<beta.get(i) << " " <<alpha.get(i) <<endl;
    return 0;
    }
```

- Output:

```
9
        8
        7
        6
        5
        4
        3
        2
        1
        0
        1 1
        2 2
        3 3
        4 4
        5 5
        6 6
        7 7
        8 8
        9 9
        10 10
```

-
```
    #include <iostream>

        using namespace std;

        class list {
          int *t;

        public:
          list() {
            t = new int[10];
```

```
    for (int i = 0; i < 10; i++)
      t[i] = i;
  }

  list(const list &obj) {  // Deep copy constructor
    t = new int[10];
    for (int i = 0; i < 10; i++)
      t[i] = obj.t[i];
  }

  ~list() {  // Destructor to free allocated memory
    delete[] t;
  }

  void put(int i, int j) {
    if (i >= 0 && i < 10)
      t[i] = j;
  }

  int get(int i) const {
    return t[i];
  }
};

int main() {
  list beta;
  int i;

  for (i = 9; i >= 0; i--)
    cout << beta.get(i) << "\n";

  list alpha(beta);

  for (i = 0; i < 10; i++)
    beta.put(i, i + 1);

  for (i = 0; i < 10; i++)
    cout << beta.get(i) << " " << alpha.get(i) << endl;

  return 0;
}
```

- Updated Output:

```
9
8
7
6
```

5
4
3
2
1
0
1 0
2 1
3 2
4 3
5 4
6 5
7 6
8 7
9 8
10 9

5. Will this code compile fine? If not, why? If yes, what will be the output? Will the compiler provide a default constructor in this case?

```
#include<iostream>
using namespace std;
class Y {
        int i;
public: Y(const Y &y) { i = y.i; }
void print() { cout << i; }};
int main() {
        Y y1;
        Y y2(y1);
        y2.print();
return 0;
}
```

   - No, the code will not compile.
   - The class Y does not have a default constructor, but an object y1 is being created without passing any arguments. So it will give error, while creating y1 object.
   - Compiler will not provide a default constructor in this case. The compiler only provides a default constructor if no constructor is explicitly defined.

6. Can other members be used in expression within brackets in the initializer list? For example using member x to initialize member y. Which member will be initialized first? x or y? What will be the values of x and y?

```
#include<iostream>
using namespace std;
class K {
        int y;
        int x;
public: K(): x(3), y(x * 2) {}
void print() {
        cout << x << " " << y << endl;
}};
int main() {
        K k1;
        k1.print();
return 0;
}
```

- Yes, members can be used in expressions within the initializer list, but only if they have already been initialized. The order of initialization follows the declaration order in the class, not the order in the initializer list. That's why member y will initialized first
- Output: 3 0

7. Create a class named DynamicArray with the following private data members: int *ptr: a pointer to an integer pointing to the first element of dynamically allocated memory using malloc() instead of new for storing integer numbers, int size: an integer indicating the number of currently stored elements in the array, int capacity: an integer representing the maximum number of elements that can be stored in the array. A constructor for allocating the dynamic memory with the given size. (Initially, size = capacity). If no size is specified while creating the object instance, the array defaults to being empty (size = capacity = 0). A Destructor for dellocating the dynamically allocated memory. A public member function - void insert(int val): Inserts an element after the end of the current size of the array. If the capacity is full, it automatically resizes itself using realloc() (If capacity = 0, update it to capacity = 1 and if capacity is non-zero, double the current capacity) to accommodate the new element. Your program should also include a main() function to create two object instances of DynamicArray named array1 of size 0 and array2 of size 10, respectively, and perform the following operations: Insert the squares of integers from 0 to 9 into array1, Insert the cubes of integers from 0 to 9 into array2, and finally, print the elements of both array1 and array2

- ```
  #include using namespace std;
  class DynamicArray
  {
  	int *ptr;
  	int size;
  	int capacity;
  	public: DynamicArray(int size=0)
  	 {
  		this->capacity = size;
  		this->size=0;
  		ptr=(int*)malloc(capacitysizeof(int));
  	}
  	void insert(int val) {
  		if (capacity==0){
  			capacity=1;
  			ptr = (int)realloc(ptr, capacitysizeof(int));
  			ptr[size] = val; size++;
  		}
  		else{
  			capacity*=2;
  			ptr = (int*)realloc(ptr, capacity*sizeof(int));
  			ptr[size] = val;
  			size++;
  		}
  	}
  	void print() {
  	   for (int i = 0; i < size; i++)
  	      cout << ptr[i] << " ";
  	   cout << endl;
  	}

  	~DynamicArray(){
  	   free(ptr);
  	}
  	};
  	int main(){
  	DynamicArray array1;
  	DynamicArray array2(10);

  	for (int i = 0; i < 10; i++)
  	   array1.insert(i*i);

  	for (int i = 0; i < 10; i++)
  	   array2.insert(i*i*i);

  	array1.print();
  	array2.print();
  ```
  ```

```
return 0;

}
```

8. Create a class named MyString with the following private data members: char *str: a pointer to a character pointing to the first element of dynamically allocated memory for storing characters, int len: an integer representing the length of the character array. A default constructor MyString(), which initializes an empty array. A parameterized constructor MyString(const char s[]), which initializes the dynamic array with the provided character array - s. A Destructor for dellocating the dynamically allocated memory. A public member function - void display(): for printing the character array. Your program should also include a main() function to create several instances of MyString objects and print the output.

➢
```
#include <iostream>
#include <cstring>
using namespace std;

class MyString {
private:
  char *str;
  int len;

public:
  MyString() {
    len = 0;
    str = new char[1];
    str[0] = '\0';
  }

  MyString(const char s[]) {
    len = strlen(s);
    str = new char[len + 1];
    strcpy(str, s);
  }

  void display() {
    cout << str << endl;
  }

  ~MyString() {
    delete[] str;
  }
```

```
};

int main() {
    MyString s1;
    MyString s2("Hello");
    MyString s3("World!");

    cout << "String 1: ";
    s1.display();

    cout << "String 2: ";
    s2.display();

    cout << "String 3: ";
    s3.display();

    return 0;
}
```