**Roll No…………….. Total No. of Pages:……**

**ST-2 (SET-IV )**

**4th SEMESTER 2022-23**

**22CS006- Object Oriented Programming**

**Time allowed: 90 Minutes Max. Marks: 40**

**General Instructions:**

* **Follow the instructions given in each section.**
* **Make sure that you attempt the questions in order.**

**SECTION-A (10\*1 mark=10 marks)**

***(All questions are compulsory)***

1. What is the access specifier used for class members to make them accessible within the class and its derived classes?
   1. public
   2. private
   3. **protected**
   4. static
2. In C++, can a derived class inherit multiple base classes?
   1. Yes, using single inheritance
   2. **Yes, using multiple inheritance**
   3. No, inheritance is limited to one base class
   4. Yes, using multilevel inheritance
3. Which of the following operator cannot be used to overload when that function is declared as friend function?
   1. -=
   2. ||
   3. ==
   4. **[]**
4. Which function is automatically called when an object is created?
   1. delete()
   2. destructor
   3. main()
   4. **constructor**
5. How do you define a constructor with no parameters in C++?
   1. **Constructor()**
   2. constructor(void)
   3. Constructor(void)
   4. Constructor::Constructor()
6. When is the destructor of a class called?
   1. When the object is created
   2. **When the object goes out of scope**
   3. When the object is passed to a function
   4. When the destructor is explicitly called
7. Can constructors be virtual in C++?
   1. Yes
   2. **No**
8. What is the syntax of overloading operator + for class A?
   1. **A operator+(argument\_list){}**
   2. A operator[+](argument\_list){}
   3. int +(argument\_list){}
   4. int [+](argument\_list){}
9. What is the return type of the copy constructor?
   1. void
   2. int
   3. **The same class object (by reference)**
   4. The same class object (by value)
10. Which of the following cannot be a friend?
    1. Function
    2. Class
    3. **Object**
    4. Operator function

**SECTION-B (5\*2 mark=10 marks)**

***(All questions are compulsory)***

1. Which of the following feature of OOPs is not used in the following C++ code?

class A{

int i;

public:

void print() {cout << "hello" << i;}

}

class B : public A{

int j;

public:

void assign (int a ) {k = a;}

}

a) Abstraction

b) Encapsulation

c) Inheritance

**d) Polymorphism**

1. What is the output of the following code?

#include <iostream>

using namespace std;

class Base {

public:

Base() { cout << "Base constructor" << endl; }

~Base() { cout << "Base destructor" << endl; }

};

class Derived : public Base {

public:

Derived() { cout << "Derived constructor" << endl; }

~Derived() { cout << "Derived destructor" << endl; }

};

int main() {

Base\* ptr = new Derived();

delete ptr;

return 0;

}

**a) Base constructor, Derived constructor, Derived destructor, Base destructor**

b) Derived constructor, Base constructor, Base destructor, Derived destructor

c) Base constructor, Derived constructor, Base destructor

d) Derived constructor, Base destructor

1. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class complex

{

int i;

int j;

public:

complex(){}

complex(int a, int b)

{

i = a;

j = b;

}

complex operator+(complex c)

{

complex temp;

temp.i = this->i + c.i;

temp.j = this->j + c.j;

return temp;

}

void show(){

cout<<"Complex Number: "<<i<<" + i"<<j<<endl;

}

};

int main(int argc, char const \*argv[])

{

complex c1(1,2);

complex c2(3,4);

complex c3 = c1 + c2;

c3.show();

return 0;

}

**a) Complex Number: 4 + i6**

b) Complex Number: 2 + i2

c) Error

d) Segmentation fault

1. What is the output of the following code?

#include <iostream>

using namespace std;

class MyClass {

public:

int x;

MyClass() : x(10) {}

};

int main() {

MyClass obj1;

MyClass obj2 = obj1;

cout << obj2.x << endl;

return 0;

}

**a) 10**

b) 0

c) Garbage value

d) Compiler error

1. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class Box

{

int capacity;

Box(){}

Box(double capacity){

this->capacity = capacity;

}

};

int main(int argc, char const \*argv[])

{

Box b1(10);

Box b2 = Box(14);

return 0;

}

**a) Error**

b) Segmentation fault

c) 4

d) No output

**SECTION-C(Coding Question) (2x5 marks=5 marks)**

Q16) You have given a string. Write a function that reverses a string using a stack data structure.

**Input :** hello

**Output:** olleh

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | JAVA | Stack Overflow | Pune |
| **Output** | AVAJ | wolfrevO kcatS | enuP |

Solution :

**#include <stdio.h>**

**#include <string.h>**

**#define MAX\_SIZE 100**

**char stack[MAX\_SIZE];**

**int top = -1;**

**void push(char ch) {**

**//check for stack is full or not**

**if (top == MAX\_SIZE - 1) {**

**printf("Stack Overflow\n");**

**return;**

**}**

**//push element to stack**

**stack[++top] = ch;**

**}**

**char pop() {**

**//check for stack is empty or not**

**if (top == -1) {**

**printf("Stack Underflow\n");**

**return -1;**

**}**

**//remove element from top of the stack**

**return stack[top--];**

**}**

**void reverseString(char\* str) {**

**int len = strlen(str);**

**//push all characters of the string one by one**

**for (int i = 0; i < len; i++)**

**push(str[i]);**

**//pop all characters of the string one by one and store in same string**

**for (int i = 0; i < len; i++)**

**str[i] = pop();**

**//after this, str will contain reversed string**

**}**

**int main() {**

**char str[100];**

**printf("Enter a string: ");**

**scanf("%[^\n]%\*c", str);**

**printf("Original string: %s\n", str);**

**reverseString(str);**

**printf("Reversed string: %s\n", str);**

**return 0;**

**}**

Q17) You are tasked with developing a program that performs number swapping using macros.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | 5 10 | 18 93 | 9 7 |
| **Output** | Before swap: x = 5, y = 10  After swap: x = 10, y = 5 | Before swap: x = 18, y = 93  After swap: x = 93, y = 18 | Before swap: x = 9, y = 7  After swap: x = 7, y = 9 |

Solution :

**#include <stdio.h>**

**#define SWAP(a, b) do { \**

**int temp = (a); \**

**(a) = (b); \**

**(b) = temp; \**

**} while (0)**

**int main() {**

**int x = 5, y = 10;**

**printf("Before swap: x = %d, y = %d\n", x, y);**

**SWAP(x, y); //cal SWAP Macro**

**printf("After swap: x = %d, y = %d\n", x, y);**

**return 0;**

**}**

**SECTION-D (Coding Question)(1x10 mark=10 mark)**

Q18) You are given an array of integers nums, there is a sliding window of size k which is moving from the very left of

the array to the very right. You can only see the k numbers in the window. Each time the sliding window moves right by one position.

Return the max sliding window.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | [1,3,-1,-3,5,3,6,7], k = 3 | [2, 3, 0, -6, 4, 3], k = 3 | [2, 3, 0, -6, 4, 3], k = 2 |
| **Output** | [3,3,5,5,6,7] | [3,3,4,4] | [3,3,0,4,4] |

Solution :

**#include<stdio.h>**

**#include<stdlib.h>**

**#define MAX\_SIZE 100**

**int adjMatrix[MAX\_SIZE][MAX\_SIZE];**

**int visited[MAX\_SIZE];**

**struct queue**

**{**

**int size;**

**int f;**

**int r;**

**int\* arr;**

**};**

**//check if queue is empty**

**int isEmpty(struct queue \*q){**

**if(q->r==q->f){**

**return 1;**

**}**

**return 0;**

**}**

**//check if queue is full**

**int isFull(struct queue \*q){**

**if(q->r==q->size-1){**

**return 1;**

**}**

**return 0;**

**}**

**//add element in queue**

**void enqueue(struct queue \*q, int val){**

**if(isFull(q)){**

**printf("This Queue is full\n");**

**}**

**else{**

**q->r++;**

**q->arr[q->r] = val;**

**// printf("Enqued element: %d\n", val);**

**}**

**}**

**//remove element from queue**

**int dequeue(struct queue \*q){**

**int a = -1;**

**if(isEmpty(q)){**

**printf("This Queue is empty\n");**

**}**

**else{**

**q->f++;**

**a = q->arr[q->f];**

**}**

**return a;**

**}**

**void BFS(int start,int numVertices){**

**// Initializing Queue (Array Implementation)**

**struct queue q;**

**q.size = 400;**

**q.f = q.r = 0;**

**q.arr = (int\*) malloc(q.size\*sizeof(int));**

**// BFS Implementation**

**int node;**

**printf("%d ", start);**

**visited[start] = 1;**

**enqueue(&q, start); // Enqueue i for exploration**

**while (!isEmpty(&q))**

**{**

**int node = dequeue(&q);**

**for (int j = 0; j < numVertices; j++)**

**{**

**if(adjMatrix[node][j] ==1 && visited[j] == 0){**

**printf("%d ", j);**

**visited[j] = 1;**

**enqueue(&q, j);**

**}**

**}**

**}**

**}**

**int main(){**

**int numVertices;**

**printf("Enter the number of vertices: ");**

**scanf("%d", &numVertices);**

**printf("Enter the adjacency matrix:\n");**

**for (int i = 0; i < numVertices; i++) {**

**for (int j = 0; j < numVertices; j++) {**

**scanf("%d", &adjMatrix[i][j]);**

**}**

**}**

**// Initialize visited array**

**for (int i = 0; i < numVertices; i++) {**

**visited[i] = 0;**

**}**

**int startVertex;**

**printf("Enter the starting vertex: ");**

**scanf("%d", &startVertex);**

**BFS(startVertex, numVertices);**

**return 0;**

**}**