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

**ST-2 (SET-VI)**

**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. Which of the following is not a type of Constructor?
   1. **Friend constructor**
   2. Copy constructor
   3. Default constructor
   4. Parameterized constructor
2. Which of the following is correct?
   1. Base class pointer object cannot point to a derived class object
   2. **Derived class pointer object cannot point to a base class object**
   3. A derived class cannot have pointer objects
   4. A base class cannot have pointer objects
3. In case of non-static member functions how many maximum object arguments a binary operator overloaded function can take?
   1. **1**
   2. 2
   3. 3
   4. 0
4. Which of the following class allows to declare only one object of it?
   1. Abstract class
   2. Virtual class
   3. **Singleton class**
   4. Friend class
5. Out of the following, which is not a member of the class?
   1. Static function
   2. **Friend function**
   3. Constant function
   4. Virtual function
6. Which of the following statements about friend classes in C++ is correct?
   1. **Friend classes have access to the private members of the class they are friends with**
   2. Friend classes are inherited along with the base class
   3. Friend classes can be used to achieve multiple inheritance
   4. Friend classes must be defined within the scope of the base class
7. Which constructor is called when an object is created using the default constructor syntax?
   1. Copy constructor
   2. Parameterized constructor
   3. Destructor
   4. **Default constructor**
8. Which of the following is a static polymorphism mechanism?
   1. Function overloading
   2. Operator overloading
   3. Templates
   4. **All of the mentioned**
9. Can a constructor be virtual in C++?
   1. Yes
   2. **No**
10. What is the purpose of the "virtual" keyword in C++?
    1. It is used to make a member variable of a class constant
    2. **It is used to allow a function in the base class to be overridden in the derived class**
    3. It is used to create a pointer to an object
    4. It is used to prevent inheritance from a class

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

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

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

#include <iostream>

using namespace std;

void print(int i)

{

cout << i;

}

void print(double f)

{

cout << f;

}

int main(void)

{

print(5);

print(500.263);

return 0;

}

* 1. **5500.263**
  2. 500.2635
  3. 500.263
  4. 500.266

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

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"Default constructor called\n";

}

A(const A& a){

cout<<"Copy Constructor called\n";

}

};

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

{

A obj;

A a1 = obj;

A a2(obj);

}

1. Default constructor called

Copy Constructor called

1. **Default constructor called**

**Copy Constructor called**

**Copy Constructor called**

1. Default constructor called

Default constructor called

Copy Constructor called

1. Copy Constructor called

Default constructor called

Copy Constructor called

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

#include <iostream>

#include <string>

using namespace std;

class A{

mutable int a;

public:

A(){

cout<<"A's Constructor called\n";

}

~A(){

cout<<"A's Destructor called\n";

}

};

class B: public A{

public:

B(){

cout<<"B's Constructor called\n";

}

~B(){

cout<<"B's Destructor called\n";

}

};

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

{

B b1;

}

a)

A's Constructor called

B's Constructor called

b)

A's Destructor called

B's Destructor called

**c)**

**A's Constructor called**

**B's Constructor called**

**B's Destructor called**

**A's Destructor called**

d)

A's Constructor called

B's Constructor called

A's Destructor called

B's Destructor called

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

#include <iostream>

#include <string>

using namespace std;

class A

{

mutable int a;

public:

int assign(int i) const {

a = i;

}

int return\_value() const {

return a;

}

};

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

{

A obj;

obj.assign(5);

cout<<obj.return\_value();

}

1. **5**
2. Error
3. Segmentation fault
4. Undefined value
5. What happens when objects s1 and s2 are added?

string s1 = "Hello";

string s2 = "World";

string s3 = (s1+s2).substr(5);

1. Error because s1+s2 will result into string and no string has substr() function
2. Segmentation fault as two string cannot be added in C++
3. **The statements runs perfectly**
4. Run-time error

**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;**

**}**