**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**RAMAPURAM CAMPUS**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**QUESTION BANK**

**18CSC202J OBJECT ORIENTED DESIGN AND PROGRAMMING**

**UNIT-1**

**1 marks**

**1. Dividing a program into functions**

A) is the key to object-oriented programming

B) makes the program easier to conceptualize

C) may reduce the size of the program

D) makes the program run faster

Reference-5

**2**. **Specify how many bytes are occupied by the following data types in a 32-bit system.**

A) Type int B) Type long double

C) Type float D)Type long

(page no 70) Reference-5

**3. A function’s single most important role is to……………………..**

A) give a name to a block of code

B) reduce program size

C) accept arguments and provide a return value

D) help organize a program into conceptual units

(page no 210) Reference-5

**4. A class consists of which of these abstractions**

A) Set of the objects

B) Operations

C) Attributes

D) All the mentioned

(Ans: d)

**5. The Unified Modeling Language is ………………………**

A) a program that builds physical models.

B) a way to look at the organization of a program

C) the combination of C++ and FORTRAN

D) helpful in developing software systems.

(Page no 27)- Reference-5

**6. Which of the following explains Polymorphism?**

A) intfunc(int, int);

Float func1(float,float);

B) intfunc(int);

Intfunc(int);

C) intfunc(float);

Intnew\_func();

D) intfunc();

Intnew\_func();

(Ans: C)

**7. In C++, a function contained within a class is called ………………..**

A) a member function

B) an operator

C) a class function

D) a method

(page no 25)

**8. What happens if the base and derived class contains definition of a function with same prototype?**

A) Compiler reports an error on compilation

B) Only base class function will get called irrespective of object

C) Only derived class function will get called irrespective of object

D) Base class object will call base class function and derived class object will call derived class object will call derived class function

(Ans:D)

**9. Which one of the following option is correct about the statement given below? The compiler checks the type of reference in the object and not the type of object**

A) inheritance

B) Polymorphism

C) Abstraction

D) Encapsulation

(Ans: B)

**10. Which of the following functions are performed by a constructor**?

A) Construct a new class

B) Construct a new object

C) Construct a new function

D) Initialize objects

Ans: D

**11. How many types of Polymorphisms are supported by C++**

A) 1 B) 2 C) 3 D) 4

Ans: B

**12. Which of the following is the correct class of the object cout**

A) iostream

B) istream

C) ostream

D) ifstream

Ans: C

**13. In UML, diagrams which captures system static structure and provide foundation for other models is called ...............**

A) Deployment diagram B) Class diagram

C) Component diagram D) Object diagram

Ans: B

**14. Which error will be produced if private members are accessed?**

A) Can’t access private message

B) Code unreachable

C) Core dumped

D) Bad code

Ans: A

**15. The default access specifier for the class member is**

A) public B) private C) protected D) None of the above

Ans: B

**16.Which of the following is CPP style type-casting?**

A) per=total/ (float)m

B) per=total/float(m)

C) per = (float)total/m

D) None of these

Ans: B

**17. What is the output of the following program?**

#include<iostream>

using namespace std;

void main()

{

char s[] = “SRM”;

\*s = ‘R’;

cout<<s<<endl;

}

1. RRM B) SRM C) SRR D) None of these

Ans: A

**18. What does the following statement mean?**

int (\*fp)(char\*)

a)pointer to a pointer  
b) pointer to an array of chars

c) pointer to function taking a char\* argument and returns an int

d) function taking a char\* argument and returning a pointer to int

Ans : c

**19.Which of the following concepts of OOPS means exposing only necessary information to client?**

A) Encapsulation B) Abstraction

C) Polymorphism C) Data binding

Ans: B

**20.  Which of the following is illegal?**a) int \*ip;  
b) string s, \*sp = 0;  
c) int i; double\* dp = &i;  
d) int \*pi = 0;

Ans: c

**21. Which member can never be accessed by inherited classes?**

A) Private member function

B) Public member function

C) Protected member function

D) All can be accessed

Ans: A

**22. What will happen in this code?**

**int a=100, b=200;**

**int \*p=&a, \*q=&b;**

**p=q;**

a) b is assigned to a  
b) p now points to b  
c) a is assigned to b  
d) q now points to a

Ans: b

**23.  The size\_t integer type in C++ is?**  
a) Unsigned integer of at least 64 bits  
b) Signed integer of at least 16 bits  
c) Unsigned integer of at least 16 bits  
d) Signed integer of at least 64 bits

Ans: c

**24. A static local variable is used to ………………….**

A) make a variable visible to several functions

B) make a variable visible to only one function

C) conserve memory when a function is not executing

D) retain a value when a function is not executing

(page no 212 Reference -5)

**25. An operation can be described as?**

A) Object behaviour B) Class behaviour

C) Functions D) a,b

Ans: D

**4 marks**

1. **List out the difference between procedure oriented programming & Object oriented programming.**

OOPs makes development and maintenance easier. But in Procedure-oriented programming language, it is not easy to manage if code grows as project size grows. OOPs provides data hiding whereas in Procedure-oriented programming language, global data can be accessed from anywhere. OOPs provides ability to simulate real-world event much more effectively. We can provide the solution of real word problem if we are using the ObjectOriented Programming language.

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Procedural Oriented programming** | **Object Oriented Programming** |
| **1** | In procedural programming, program is divided into small parts called **functions**. | In object oriented programming, program is divided into small parts called **objects**. |
| **2** | Procedural programming follows **top down approach**. | Object oriented programming follows **bottom up approach**. |
| **3** | There is no access specifier in procedural programming. | Object oriented programming have access specifiers like private, public, protected etc. |
| **4** | Adding new data and function is not easy. | Adding new data and function is easy. |
| **5** | Procedural programming does not have any proper way for hiding data so it is **less secure**. | Object oriented programming provides data hiding so it is **more secure**. |
| **6** | In procedural programming, overloading is not possible. | Overloading is possible in object oriented programming. |
| **7** | In procedural programming, function is more important than data. | In object oriented programming, data is more important than function. |
| **8** | Procedural programming is based on **unreal world**. | Object oriented programming is based on **real world**. |
| **9** | Examples: C, FORTRAN, Pascal, Basic etc. | Examples: C++, Java, Python, C# etc. |

1. **What is scope of the variable? Illustrate with an example?**

**Local Variables**

Local variables are the variables which exist only between the curly braces, in which its declared. Outside that they are unavailable and leads to compile time error.

*Example*:

include<iostream>

using namespace std;

int main()

{

int i=10;

if(i<20) // if condition scope starts

{

int n=100; // Local variable declared and initialized

} // if condition scope ends

cout<< n; // Compile time error, n not available here

}

**Global variables**

Global variables are those, which ar once declared and can be used throughout the lifetime of the program by any class or any function. They must be declared outside the main() function. If only declared, they can be assigned different values at different time in program lifetime. But even if they are declared and initialized at the same time outside the main() function, then also they can be assigned any value at any point in the program.

For example: Only declared, not initialized

include<iostream>

using namespace std;

int x; // Global variable declared

int main()

{

x=10; // Initialized once

cout<<"first value of x = "<< x;

x=20; // Initialized again

cout<<"Initialized again with value = "<< x;

}

1. **Write about class, objects, methods & messages.**

**Class**

A class is a blueprint for the object.

classclassName

{

// data members

// member functions

};

**Objects**

Object is an instance of a class

When class is defined, only the specification for the object is defined; no memory or storage is allocated.

To use the data and access functions defined in the class, you need to create objects.

classNameobjectVariableName;

**Method:**

A method is an action which an object is able to perform.

1. **Write a C++ program to generate Fibonacci series using class**

#include<iostream>

using namespace std;

class Test {

public:

intfibo(int x) {

int a=-1,b=1,c;

for (i = 1; i <= x; i++) {

c=a+b

cout<<” “<<c;

a=b

b=c

}

}

};

int main() {

int x;

cout<< "Enter a Number :";

cin>> x;

Test obj;

obj.fibo(x);

return 0;

}

1. **Write a c++ program to find factorial of a given number using class**

#include<iostream>

using namespace std;

class Test {

public:

int factorial(int x) {

int i, f = 1;

for (i = 1; i <= x; i++) {

f = f\*i;

}

return f;

}

};

int main() {

int x, f;

cout<< "Enter a Number :";

cin>> x;

Test obj;

f = obj.factorial(x);

cout<< "\nThe Factorial is :" << f;

return 0;

}

1. **What is static variable?**

Variable declared with keyword static is a static variable. It is a class level variable commonly shared by all objects of the class.

* Memory allocation for such variables only happens once when the class is loaded in the memory. scope of the static variable is class scope ( accessible only inside the class)
* lifetime is global
* Automatically initialized to 0.
* Example: static int no;

1. **What is constructor? State the rules of constructor**

A constructor is a member function of a class which initializes objects of a class. In C++, Constructor is automatically called when object(instance of class) create. It is special member function of the class.

Rules:

* Constructor has same name as the class itself
* Constructors don’t have return type
* A constructor is automatically called when an object is created.
* If we do not specify a constructor, C++ compiler generates a default constructor for us (expects no parameters and has an empty body).

1. **Give example for static keyword and explain**

When a variable is declared as static, space for **it gets allocated for the lifetime of the program**. Even if the function is called multiple times, space for the static variable is allocated only once and the value of variable in the previous call gets carried through the next function call. This is useful for implementing [coroutines in C/C++](https://www.geeksforgeeks.org/coroutines-in-c-cpp/) or any other application where previous state of function needs to be stored.

#include <iostream>

#include <string>

using namespace std;

void demo()

{

    // static variable

    staticint count = 0;

    cout<< count << " ";

    // value is updated and

    // will be carried to next

    // function calls

    count++;

}

int main()

{

    for (int i=0; i<5; i++)

        demo();

    return 0;

}

**Output: 0 1 2 3 4**

1. **Differentiate private and public access specifier with an example**

A **public** member is accessible from anywhere outside the class but within a program. You can set and get the value of public variables without any member function as shown in the following example

|  |
| --- |
| #include<iostream>  using namespace std;    // class definition  class Circle  {      public:          double radius;            double  compute\_area()          {              return 3.14\*radius\*radius;          }    };    // main function  int main()  {      Circle obj;        // accessing public datamember outside class      obj.radius = 5.5;        cout<< "Radius is:" <<obj.radius<< "\n";      cout<< "Area is:" <<obj.compute\_area();      return 0;  } |

Output:

Radius is:5.5

Area is:94.985

**Private**

The class members declared as **private** can be accessed only by the functions inside the class. They are not allowed to be accessed directly by any object or function outside the class. Only the member functions or the [friend functions](https://www.geeksforgeeks.org/friend-class-function-cpp/) are allowed to access the private data members of a class

#include<iostream>

using namespace std;

class Circle

{

    // private data member

    private:

        double radius;

    // public member function

    public:

        double  compute\_area()

        {   // member function can access private

            // data member radius

            return 3.14\*radius\*radius;

        }

};

// main function

int main()

{

    // creating object of the class

    Circle obj;

    // trying to access private data member

    // directly outside the class

    obj.radius = 1.5;

    cout<< "Area is:" <<obj.compute\_area();

    return 0;

}

1. **What is pointer? Give an example program**

Pointers are symbolic representation of addresses. They enable programs to simulate call-by-reference as well as to create and manipulate dynamic data structures. It’s general declaration in C/C++ has the format:

datatype \*var\_name;

int \*ptr; //ptr can point to an address which holds int data

**Example Program:**

#include <bits/stdc++.h>

using namespace std;

void point()

{

    intvar = 20;

        //declare pointer variable

    int \*ptr;

    //note that data type of ptr and var must be same

    ptr = &var;

    // assign the address of a variable to a pointer

    cout<< "Value at ptr = " <<ptr<< "\n";

    cout<< "Value at var = " <<var<< "\n";

    cout<< "Value at \*ptr = " << \*ptr<< "\n";

}

//Driver program

int main()

{

    point();

}

**12 marks**

* 1. Explain in detail about all the features of OOPS with suitable example
  2. Explain in detail about constructor and demonstrate with an example
  3. Explain in detail about destructor and demonstrate with an example
  4. Explain about the types of access specifiers with an example program
  5. Explain about the class diagram with all its components
  6. Explain about use case, scenario, actors with an example
  7. Write the problem statement for Library Management system. Design UML Class diagram and explain its components
  8. Explain about the inline and friend function with an example.

**---------------------------------------------------------------------------------------------------------------------**