VIGNAN'S
Foundation for Science, Technology & Research
(Deemed to be University)

Scheme of Valuation – March / April 2024

Course	B.Tech./BBA/BCA/BSc/B.Pharmacy/MBA/MCA/M.Tech/Ph.D/DIPLOMA				
Regulation	R22	Year	II	Semester	I
Branch / Specialization	DIPLOMA in CSE		Subject Code	22DCS203	
Subject Name	OBJECT ORIENTED PROGRAMMING through C++				
Scheme Prepared by	Dr. / Mr. / Ms.: A. SURESH BABU Designation: ASSISTANT PROFESSOR Department: DCSE				
Signature	A. Sureshbabu				
Date of Exam	06-04-2024				
Scheme Submission Date	06-04-2024				

START HERE

OBJECT ORIENTED PROGRAMMING THROUGH C++

Scheme of Valuation

SECTION-A

10×1M=10 M

1—b

2—c

3—b

4—b

5—d

6—d

7—c

8—b

9—c

10—b

SECTION—B

4x5=20M

11 List of operators---2M

Explanation—3M

Ans:

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Bitwise Operators
- Assignment Operators
- Ternary or Conditional Operators

OR

.

12. Structure of C++ Program-----3M

Example---2M

Ans:

Standard Libraries Section

Main Function Section

Function Body Section

Standard libraries section:

- **#include** is a specific preprocessor command that effectively copies and pastes the entire text of the file, specified between the angle brackets, into the source code.
- The **file <iostream>,** which is a standard file that should come with the C++ compiler, is short for input-output streams. This command contains code for displaying and getting an input from the user.
- **namespace** is a prefix that is applied to all the names in a certain set. **iostream** file defines two *names* used in this program - cout and endl.
- This code is saying: Use the cout and endl tools from the std toolbox.

Main Function Section

- The starting point of all C++ programs is the main function.
- This function is called by the operating system when your program is executed by the computer.
- { signifies the start of a block of code, and } signifies the end

Function body Section

- The name **cout** is short for **character output** and displays whatever is between the << brackets.
- Symbols such as << can also behave like functions and are used with the keyword **cout.**
- The **return** keyword tells the program to return a value to the function **int main**
- After the return statement, execution control returns to the operating system component that launched this program.
- Execution of the code terminates here.

13. Distinguish between call by value and call by address----3M

Example---2M

Ans

Aspect	Call by Value	Call by Reference
What is passed	Copies the actual value of the	Passes a reference or

	argument.	memory address of the argument.
Modification of argument	Changes made to the parameter inside the function do not affect the original argument	Changes made to the parameter inside the function directly affect the original argument.
Memory Usage	Typically consumes less memory as it creates a separate copy of the argument.	May consume more memory as it uses references to the original data.
Performance	Generally faster due to the smaller data transfer.	May be slower due to the need to access data through references.
Data Type Constraints	Suitable for simple data types (int, float, etc.).	Suitable for complex data types (arrays, objects, etc.).
Risk of unintended changes	Reduces the risk of unintentional side effects.	Increases the risk of unintentional side effects.

OR

14. **Function overloading—2M**

Principles of function overloading—3M

Ans: **Function overloading**: Function overloading is a feature of object-oriented programming where two or more functions can have the same name but different parameters. When a function name is overloaded with different jobs it is called Function Overloading. In Function Overloading "Function" name should be the same and the arguments should be different

Principles of function overloading:

- The functions must have the same name.
- The functions must have different types of parameters.
- The functions must have a different set of parameters.
- The functions must have a different sequence of parameters.

15. Constructor overloading—2M
Example—3M

Constructor overloading:

Constructor overloading means having more than one constructor with the same name. Constructors are methods invoked when an object is created. You have to use the same name for all the constructors which is the class name. This is done by declaration the constructor with a different number of arguments

OR

16. Operator overloading—2M

Example---3M

Operator overloading:

C++ has the ability to provide the operators with a special meaning for a data type, this ability is known as operator overloading. Operator overloading is a compile-time polymorphism. For example, we can overload an operator '+' in a class like String so that we can concatenate two strings by just using +

17 Virtual function—2M

Rules for Virtual functions—3M

Ans:

Virtual function : A virtual function in C++ is a base class member function that you can redefine in a derived class to achieve polymorphism. You can declare the function in the base class using the virtual keyword

Rules for Virtual functions:

1. Virtual functions cannot be static.
2. A virtual function can be a friend function of another class.
3. Virtual functions should be accessed using a pointer or reference of base class type to achieve runtime polymorphism.
4. The prototype of virtual functions should be the same in the base as well as the derived class.
5. They are always defined in the base class and overridden in a derived class. It is not mandatory for the derived class to override (or re-define the virtual function), in that case, the base class version of the function is used.
6. A class may have a **virtual destructor** but it cannot have a virtual constructor.

OR

18 Any five differences between class template and function template—5x1=5M

SECTION – C

19. OOP—2M

Any 4 oops principles—4x2=8M

Ans:

Object-oriented programming: Object-oriented programming (OOP) is a style of programming characterized by the identification of classes of objects closely linked with the methods (functions) with which they are associated.

OOPs principles:

1. Class
2. Objects
3. Encapsulation
4. Abstraction
5. Polymorphism
6. Inheritance
7. Dynamic Binding
8. Message Passing

20. **Inline function—2M**
    Program for finding the area of a triangle using inline functions—8M

    Ans:

    Inline function: An inline function is one for which the compiler copies the code from the function definition directly into the code of the calling function rather than creating a separate set of instructions in memory.

    Program for finding the area of a triangle using inline functions

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

// inline function, no need prototype
inline float triangle_area(float base, float height)
{
float area;
area = (0.5 * base * height);
return area;
}

int main(void)
{
float b, h, a;
b = 4;
h = 6;
// compiler will substitute the inline function code here.
a = triangle_area(b, h);
cout<<"Area = (0.5*base*height)"<<endl;
cout<<"where, base = 4, height = 6"<<endl;
cout<<"\nArea = "<<a<<endl;

return 0;
}
```

21 **C++ program to overload binary '+' operator which will add two complex numbers of the form (a+ib).—10M**

```
#include <iostream>
#include <sstream>
#include <cmath>
using namespace std;
class Complex {
   private:
      int real, imag;
   public:
   Complex(){
      real = imag = 0;
   }
   Complex (int r, int i){
      real = r;
      imag = i;
   }
   string to_string(){
      stringstream ss;
      if(imag >= 0)
         ss << "(" << real << " + " << imag << "i)";
      else
         ss << "(" << real << " - " << abs(imag) << "i)";
      return ss.str();
   }
   Complex operator+(Complex c2){
      Complex ret;
      ret.real = real + c2.real;
      ret.imag = imag + c2.imag;
      return ret;
   }
};
int main(){
   Complex c1(8,-5), c2(2,3);
   Complex res = c1 + c2;
   cout << res.to_string();
}
```

22. **C++ program to show the functioning of virtual functions—10M**

23. Inheritance—2M

Different types of inheritances used in C++----8M

Ans:

Inheritance:

Inheritance is a method through which one class inherits the properties from its parent class. Inheritance is a feature in which one new class is derived from the existing ones. The new class derived is termed a derived class, and the current class is termed a Parent or base class

Different types of inheritances used in C++:

Single Inheritance

Multiple Inheritance

Multilevel Inheritance

Hierarchical Inheritance

Hybrid Inheritance

24. **Template—2M**

Need of Template—3M

How to Declare a Template class ---5M

Ans:

Template:
Templates in c++ is an interesting feature that is used for generic programming and templates in c++ is defined as a blueprint or formula for creating a generic class or a function.

Need of template:

Templates simplify the creation of documents. Templates can ease our workload and make us feel less stressed, and, at the same time, they increase efficiency. Templates increase the attention of the audience