C++ Concepts
(Differences) A-Z

A.

Fundamental Data Types	Derived Data Types
• Basic Data Type	• Derived from Basic Data types
• Int,char,void, float,double	• Array,struct,class

B.

Struct	Class
• Members are Public by default	Members are Private by default

C.

Dynamic Memory Allocation	Static Memory Allocation
• Allocated during Run-time and the allocated memory to a variable can be altered/deallocated anytime during the run time	Also Allocated during program but the memory allocated is fixed and is de-allocated after the program is over.
• Memory can be deallocated during run time.	Memory can only be deallocated when lifetime of the corresponding variable gets over.

D.

Call by Reference	Call By value
• Value gets reflected at original location.	Value of the corresponding actual parameter remains unchanged.
• The actual parameters can be variables only	The actual parameters can be any expressions – variables, constants, or expressions

E.

Global Variables	Local Variables
Variables defined above all function definitions	Variables defined within a block or a function.
Their scope is the whole program	Their scope is only the block in which they are defined

F.

#define Macro	Function
Code gets substituted at place of function call.	Memory control is transferred at the place of function defined.

G.

= Assignment Operator	= = Comparison Operator

Assigns a value to the variable.	Compares two values and returns 1 or 0.

H.

Logical Operators	Relational Operators
! Not, && AND, || OR	< ,>, == , !=,>=,<=

I

Compile Time Errors	Run Time Errors
Syntax errors occur at compile time.	Run Time errors occur during the program execution.

J

Member Functions	Non member Functions
➢ Defined/Declared inside the class	➢ Defined Outside the class
➢ Public Member Functions are accessed by object of that class	➢ Functions are called by their name and object can passed as parameters.

K.

Break	Continue
➢ Takes the control out of the loop	Takes the control back to next iteration.

L

Switch Case	If..else
➢ Only used with char / int type of expressions ➢ Only used for equality comparison	➢ Can be used with all data types ➢ Can be used for equality comparison as well as for range checking.

M

Text Files	Binary Files
➢ Data in ASCII format ➢ Not Secure and contains plain text	➢ Data in Binary form ➢ Data stored in blocks of object size.

N

File Pointer Position	Opening Mode

Beginning	Ios::in, ios::out, ios::app
End	Ios::ate

O

Seeekg/Seekp	Tellg/tellp
Place the file pointer at desired position	Tell the current position of pointer

P

Statement	Placement of file pointer
➢ f.seekg(0);	Beginning
➢ f.seekg(40);	40 bytes ahead from beginning
➢ f.seekg(0,ios::end)	End of file
➢ f.seekg(-10,ios::cur)	10 bytes back from current position.

Q

Stack	Queue
➢ LIFO Manner	➢ FIFO Manner
➢ Only one open end – Top	Two open ends ➢ Front – for Deletion ➢ Rear – for Insertion

R

Constructor	Destructor
➢ Automatically called when object is declared	➢ Automatically called when object lifetime is over.
➢ Can be overloaded.	➢ Can't be overloaded.

S

char *str="Computer";	char s[]="Computer";
1. sizeof(str) = 2 2. strlen(str) = 8	1. sizeof(str) = 9 2. strlen(str) = 8

T.

Multilevel Inheritance	Multiple Inheritance

A → B → C	A B \ / C

U.

While	Do..while
Entry control loop	Exit control loop
Does not execute even once if the condition is false in the beginning	Executes atleast once even if the condition is false.

V

f.read((char*)&obj,sizeof(obj));	f.write((char*)&obj,sizeof(obj));
Two parameters : (char*)&obj – explicit typecasting, converting object into string of size of object passed as parameter 2 , sizeof(obj) and reads from file and stores into object.	Two parameters : (char*)&obj – explicit typecasting, converting object into string of size of object passed as parameter 2 , sizeof(obj) and writes object to file .

f.read((char*)&obj,sizeof(obj));	f.write((char*)&obj,sizeof(obj));
Two parameters : • (char*)&obj – explicit typecasting, converting object (obj) read from the file into a string • sizeof(obj) specifies the number of bytes to be read from the file	Two parameters : • (char*)&obj – explicit typecasting, converting object (obj) to be written to file into string • sizeof(obj) specifies number of bytes to be written to file.

W

Function Prototype	Function Defintion
Function Header with list of parameters passed , return type mentioned , ended with a ;.	Complete function containing header and body/statements to be executed. The header must match with Function header of defined body.

X

Reference Variable	Typedef
Alias of a Variable . int &ch=a; Ch is alias of a.	Typedef gives an alias to a datatype. Typedef float amount;

Y

Function Overloading	Function Overriding
Overloading - Two functions having same name and return Type, but with different type and/or number of arguments.	Overriding - When a function of base class is re-defined in the derived class.

Z

Arrays	Pointers
Array –array use subscripted [] variable to access and manipulate the data ,array variables can be equivalently written using pointer expression	Pointer –pointer is a variable that holds the address of variable memory location .It is used to manipulate data using the memory address. pointers use the * operator (dereference operator)to access the data pointed by them

EXPECTED VIVA questions(SOLVED)

1. What is a class?

Class is concrete representation of an entity. It represents a group of objects, which hold similar attributes and behavior. It provides Abstraction and Encapsulations. Classes are generally declared using the keyword `class`.

2. What is an Object? What is Object Oriented Programming?

Object represents/resembles a Physical/real entity. An object is simply something you can give a name. Object Oriented Programming is a Style of programming that represents a program as a system of objects and enables code-reuse.

3. What is Encapsulation?

Encapsulation is binding of attributes and behaviors. Hiding the actual implementation and exposing the functionality of any object. Encapsulation is the first step towards OOPS, is the procedure of covering up of data and functions into a single unit (called class). Its main aim is to protect the data from outside world

4. What is Abstraction?

Hiding the complexity. It is a process of defining communication interface for the functionality and hiding rest of the things.

5. What is functions Overloading?

Adding a new method with the same name in same/derived class but with different number/types of parameters. It implements Polymorphism.

6. What is Inheritance?

It is a process in which objects of one class acquire the properties of object of another class.

7. What is an Abstract class?

An abstract class is a special kind of class that cannot be instantiated. It normally contains one or more abstract methods or abstract properties. It provides body to a class.

8. What is Polymorphism? And its type?

It is the ability for a message or data to be processed in more than one form. Polymorphism is a property by which the same message can be sent to objects of several different classes. Polymorphism is implemented in C++ through virtual functions and overloading- function overloading and operator overloading.

1. **What is inheritance and its type?**
2. **What is the difference b/n public, private and protected?**

- **Public:** The data members and methods having public as access outside the class.
- **Protected:** The data members and methods declared as protected will be accessible to the class methods and the derived class methods only.
- **Private:** These data members and methods will be accessible not from outside the class.

3. **What is a void return type?**

A void return type indicates that a method does not return a value.

4. **What is the difference between a while statement and a do statement?**

A while statement checks the loop condition at the beginning of a loop to see whether the next loop iteration should occur. A do statement checks the loop condition at the end of a loop to see whether the next loop iteration should occur.

5. **What is preprocessor?**

The preprocessor is used to handle directives for source file inclusion (#include) or defining macro definitions (#define).

Example:

```
#include <iostream.h>
#include <conio.h>
```

#define

It is used to define a macro or give name to a symbolic constant. The macro substitution is done during compile time.

Example:

```
#define MAX 80 // gives name to symbolic
constant
#define Area(L,B) L*B यmacro
void main () {
int a,b,ar; cin>>a>>b;
(a<b)?a=MAX:b=MAX;
ar=Area(a,b) ; cout<<ar<<endl;}
```

6. **What are memory management operators?**

There are two types of memory management operators in C++:

- **new**
- **delete**

Constructors

A special function Always called whenever an instance of the class is created.

- Same name as class name
- No return type
- Automatically called when object of class is created
- Used to allocate resources to the objects and may be used to initialize the members of class
- class Test

```
{  int a,b;
 Test()
  {   a=9;b=8;   } };
```

Here Test() is the constructor of Class Test.

7. What is copy constructor?

Constructor which initializes it's object member variables (by shallow copying) with another object of the same class. If you don't implement one in your class then compiler implements one for you.

for example:

- ○ Test t1(10); // calling Test constructor

  Test t2(t1); // calling Test copy constructor

  Test t2 = t1;// calling Test copy constructor

- Copy constructors are called in following cases:
- when a function returns an object of that class by value
- when the object of that class is passed by value as an argument to a function
- when you construct an object based on another object of the same class

8. What is default Constructor?

Constructor with no arguments or all the arguments has default values. In Above Question Test() is a default constructor

9. What is a scope resolution operator?

A scope resolution operator (::), can be used to define the member functions of a class outside the class.

10. What are the advantages of inheritance?

It permits code reusability. Reusability saves time in program development. It encourages the reuse of proven and debugged code, -

11. What is difference between a queue and a circular queue?

In case of normal queue it shows overflow error if r reaches to size-1 count , even if there are empty cells in the queue. But in case of circular queue it shows the overflow condition in case all the elements are full.

MORE Questions (Answer to the point)

1. What is inheritance?
2. What is Polymorphism?
3. Is class an Object? Is object a class?
4. Why destructors invoke in reverse order?
5. What is role of constructor?
6. Why we need constructors?
7. What property of OOP is implemented in Constructors?
8. Can destructors be overloaded Yes/No & Why?

9. Can constructors be overloaded Yes/No & Why?
10. What is difference between default constructor and constructor with default arguments?
11. Is any value returned by Constructors?
12. Why the reference of an object is passed in copy constructor? What will happen if the value, no the reference, is passed?
13. When is copy constructor invoked?
14. From the given conditions (1) Sample S1=S2; (2) S1=S2 ; When will copy constructor be invoke.
15. if a derived class has no parameters for its constructor but a base class has parameterized constructor , how the constructor for the derived class would be defined?
16. Difference between for and while loops.