1. What is a class?

When you define a class, you define a blueprint for a data type. This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

A class definition starts with the keyword class followed by the class name; and the class body, enclosed by a pair of curly braces. A class definition must be followed either by a semicolon or a list of declarations

2. How does a class accomplish abstraction, encapsulation, and data hiding?

abstraction is a process where you show only "relevant" data and "hide" unnecessary details of an object from the user. Consider your mobile phone, you just need to know what buttons are to be pressed to send a message or make a call, What happens when you press a button, how your messages are sent, how your calls are connected is all abstracted away from the user.

Encapsulation is the process of combining data and functions into a single unit called class. In Encapsulation, the data is not accessed directly; it is accessed through the functions present inside the class. In simpler words, attributes of the class are kept private and public getter and setter methods are provided to manipulate these attributes. Thus, encapsulation makes the concept of data hiding possible

3. What is the relationship between an object and a class?

A class provides the blueprints for objects, so basically an object is created from a class. We declare objects of a class with exactly the same sort of declaration that we declare

variables of basic types. Following statements declare two objects of class Box

4. In what way, aside from being functions, are class function members different from class data members?

the Data Members:

The public data members of objects of a class can be accessed using the direct member access operator (.).

Class member functions:

A member function of a class is a function that has its definition or its prototype within the class definition like any other variable. It operates on any object of the class of which it is a member, and has access to all the members of a class for that object.

#include <iostream>

```
#include "ex1BankAccount.h"

int main() {
  using std::cout;

  cout << "Using constructors to
creat new objects\n";
  BankAccount b1("JNU", "234",
12345.0);
  b1.show();

  BankAccount
b2=BankAccount("SCAU", "0123",
23456.0);
  b2.show();

  cout << "Deposite money to b1\n";
  b1.deposite(100.2);
  b1.show();

  cout << "withdraw money from
b2\n";
  b2.withdraw(50.1);
  b2.show();

  return 0;
}
```

6. When are class constructors called? When are class destructors called?

A class constructor is a special function in a class that is called when a new object of the class is created. A destructor is also a special function which is called when created object is deleted

8. What is a default constructor? What is the advantage of having one?

In computer programming languages the term default constructor can refer to a constructor that is automatically generated by the compiler in the absence of any programmer-defined constructors (e.g. in Java), and is usually a nullary constructor. In other languages (e.g. in C++) it is a constructor that can be called without having to provide any arguments, irrespective of whether the constructor is auto-generated or user-defined. Note that a constructor with formal parameters can still be called without arguments ifdefault arguments were provided in the constructor's definition.

10. What are this and *this?

'this'pointer in C++

The ‘this’ pointer is passed as a hidden argument to all nonstatic member function calls and is available as a local variable within the body of all nonstatic functions. ‘this’ pointer is a constant pointer that holds the memory address of the current object. ‘this’ pointer is not available in static member functions as static member functions can be called without any object (with class name).
For a class X, the type of this pointer is ‘X* const’. Also, if a member function of X is declared as const, then the type of this pointer is ‘const X *const’ (see this GFact)