UNIT I DATA ABSTRACTION & OVERLOADING

Overview of C++ – Structures – Class Scope and Accessing Class Members – Reference Variables – Initialization – Constructors – Destructors – Member Functions and Classes – Friend Function – Dynamic Memory Allocation – Static Class Members – Container Classes and Integrators – Proxy Classes – Overloading: Function overloading and Operator overloading.

1. Distinguish between Procedure Oriented Programming and Object Oriented Programming

S.No	Procedure Oriented programming	Object Oriented Programming
1.	Emphasis is on algorithm.	Emphasis is on data rather than procedure Large
2.	Programs are divided into smaller programs called functions.	Programs are divided into objects.
3.	Functions share global data.	Functions that operate on the data of an object are tied together.
4.	Data move openly around the system from function to function.	Data is hidden and cannot be accessed by external functions.
5.	Employs top-down approach in program design.	Follows bottom-up approach

2. Define Object Oriented Programming (OOP).

Object Oriented Programming is an approach that provides a way of modularizing programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand.

3. Write any four features of OOPS.

•Emphasis is on data rather than on procedure.

•Programs are divided into objects.

•Data is hidden and cannot be accessed by external functions.

•Follows bottom-up approach in program design.

4. What are the basic concepts of OOS? (DEC 2007)

- •Objects.
- •Classes.
- • Data abstraction and Encapsulation.
- •Inheritance.
- •Polymorphism.
- •Dynamic binding.
- •Message passing.

5. What are objects? How are they created? (DEC 2005) (NOV/DEC 2012)

Objects are basic run-time entities in an object-oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program has to handle. Each object has the data and code to manipulate the data and theses objects interact with each other.

Objects are created by using the **syntax:**

```
classname obj1,obj2,...,objn;
(or) during definition of the class:
class classname
{
  -------
  -------
}obj 1,obj 2,...,obj n;
```

6. What is a class? (DEC 2005)(NOV/DEC 2012)

The entire set of data and code of an object can be made a user-defined data type with the help of a class. Once a class has been defined, we can create any number of objects belonging to the classes. Classes are user-defined data types and behave like built-in types of the programming language.

7. Define Encapsulation and Data Hiding. (DEC 2005)(NOV/DEC 2011)(NOV/DEC 2013)

The wrapping up of data and functions into a single unit is known as data

encapsulation. Here the data is not accessible to the outside world. The insulation of data from direct access by the program is called data hiding or information hiding.

8. What are data members and member functions?

Classes use the concept of abstraction and are defined as a list of abstract attributes such as size, weight, and cost and uses functions to operate on these attributes. The attributes are sometimes called as data members because they hold information. The functions that operate on these data are called as methods or member functions. Eg: int a,b; // a,b are data members Void getdata ( ) ; // member function

9. Give any four advantages/benefits of OOPS. (MAY 2007)

- The principle of data hiding helps the programmer to build secure programs that cannot be invaded by code in other parts of the program.
- It is possible to have multiple instances of an object to co-exist without any interference.
- Object oriented programming can be easily upgraded from small to large systems.
- Software complexity can be easily managed.

10. What are the features required for object-based programming Language?

- Data encapsulation.
- Data hiding and access mechanisms.
- Automatic initialization and clear up of objects.
- Operator overloading.

11. Give any four applications of OOPS (MAY 2007)

- Real-time systems.
- Simulation and modeling.
- Object-oriented databases.
- AI and expert systems.

12. Give any four applications of c++?

- •Since c++ allows us to create hierarchy-related objects, we can build special object-oriented libraries, which can be used later by many programmers.
- •C++ easily maintainable and expandable.
- •C part of C++ gives the language the ability to get close to the machine-level details.
- •It is expected that C++ will replace C as a general-purpose language in the near future.

13. What are tokens?

The smallest individual units in a program are known as tokens. C++ has the following tokens, Keyword, Identifiers, Constants, Strings, Operator.

14. What are keywords?

The keywords implement specific C++ language features. They are explicitly reserved identifiers and cannot be used as names from the program variables or other user defined program elements.

Eg: goto, If, struct, else, union etc.

15. Rules for naming the identifiers in C++.

- •Only alphabetic characters, digits and underscore are permitted.
- •The name cannot start with a digit.
- •The upper case and lower case letters are distinct.
- •A declared keyword cannot be used as a variable name.

16. What are the operators available in C++?

All operators in C are also used in C++. In addition to insertion operator << and extraction operator >> the other new operators inC++ are,

- :: Scope resolution operator
- : : * Pointer-to-member declarator
- ->* Pointer-to-member operator

- .* Pointer-to-member operator
- delete Memory release operator
- endl Line feed operator
- new Memory allocation operator
- setw Field width operator

17. What is a scope resolution operator? What is the use of scope resolution operator (DEC 2007) (APRIL/MAY 2010)

Scope resolution operator is used to uncover the hidden variables. It also allows access to global version of variables.

Eg:

```
#include<iostream.h>
int m=10; // global variable m
void main ( ){int m=20; // local variable m
cout<<"m="<<m<<"\n";
cout<<": : m="<<: : m<<"\n";}
```

OUTPUT:

20

10

(: : m access global m)Scope resolution operator is used to define the function outside the class.

Syntax: return type <class name> : : <function name> **Eg:** void x : : getdata()

18. What is meant by an expression? (APRIL/MAY 2008)

An expression is a combination of constant, variable, operators and function calls written in any form as per the syntax of the c++ language.

19. Define abstraction. (DEC 2005)

Creation of well-defined interface for an object, separate from its implementation. E.g., key functionalities (init, add, delete, count, print) which can be called independently of knowing how an object is implemented.

20. What is member-dereferencing operator?

C++ permits to access the class members through pointers. It provides three pointer-to-member operators for this purpose,

- : :* To declare a pointer to a member of a class.
- To access a member using object name and a pointer to the member
- ->* To access a member using a pointer to the object and a pointer to that member.

21. What is function prototype?

The function prototype describes function interface to the compiler by giving details such as number, type of arguments and type of return values Function prototype is a declaration statement in the calling program and is of the following

data type function_name(argument list);

Eg float volume(int x,float y);

22. What is an inline function ?(NOV/DEC 2011) (MAY/JUNE 2013)

An inline function is a function that is expanded in line when it is invoked. That is compiler replaces the function call with the corresponding function code. The inline functions are defined as **Inline function-header {function body}**

Eg:

```
// Program - 4.16
// inline functions

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

inline float convert_feet(int x)
{
  return x * 12;
}

void main()
{
  clrscr();
  int inches = 45;
  cout << convert_feet(inches);
  getch();
}
```

```
// working of Program - 4.16
// inline functions

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


void main()
{
  clrscr();
  int inches = 45;
  cout << inches * 12 ;
  getch();
}
```

23. Write some situations where inline expansion may not work

For functions returning values, if loop, a switch, or a goto exists for functions not returning values , if a return statement exists if function contain static variables, if inline functions are recursive.

24. What is a default argument?

Default arguments assign a default value to the parameter, which does not have matching argument in the function call. Default values are specified when the function is declared.Eg : float amount(float principle, int period, float rate=0.15)Function call is Value=amount(5000,7);Here it takes principle=5000& period=7And default value for rate=0.15Value=amount(5000,7,0.34)Passes an explicit value 0f 0.34 to rate.We must add default value from right to left.

25. What are constant arguments?

Keyword is const. The qualifier const tells the compiler that the function should not modify the argument. The compiler will generate an error when this condition is violated. This type of declaration is significant only when we pass arguments by reference or pointer . **eg:** int strlen (constchar *p);

26. How the class is specified?(OR) How a class declared in c++? (APRIL/MAY 2010)

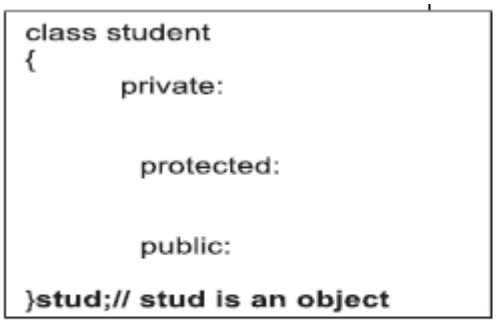
Generally class specification has two parts class declaration It describes the type and scope of its member class function definition .It describes how the class functions are implemented.

The general form is Class

```
class_name
{
private:
variable declarations;
function declaration;
public:
variable declaration;
function declaration;      };
```

27. How to create an object?

The declaration of an object is similar to that of a variable of any basic type. Objects can also be created by placing their names immediately after the closing brace of the class declaration.



```
class student
{
        private:

          protected:

          public:
}stud;// stud is an object
```

```
class student
{
        private:
        ........
        protected:
        .......
         public:
};
void main()
{
  student s, s1[5];
}
```

the variables s and s1 are objects or instances of the class stduent

28. How to access a class member?

Object-name is used to access a class member.

Function-name (actual arguments);

Eg: x.getdata (100,75.5);

29. How the member functions are defined?

Member functions can be defined in two ways outside the class definition Member function can be defined by using scope resolution operator ::

General format is

return type class_Name :: function-name(argument declaration){}

Inside the class definition This method of defining member function is to replace the function declaration by the actual function definition inside the class. It is treated as inline function.

Eg:

```
class item
{
int a,b;
void getdata(int x,int y)
```

```
{
a=x;
b=y;
};
```

30. What is static data member?

Static variable are normally used to maintain values common to the entire class.Feature:It is initialized to zero when the first object is created. No other initialization is permitted only one copy of that member is created for the entire class and is shared by all the objects It is only visible within the class, but its life time is the entire class type and scope of each static member variable must be defined outside the class .

It is stored separately rather than objects:

```
static int count//count is initialized to zero when an object is created.
int classname::count;//definition of static data member
```

31. What is static member function?

A member function that is declared as static has the following properties static function can have access to only other static member declared in the same class A static member function can be called using the class name as follows, **classname ::function_name;**

32. Define class and object (DEC 2005) (MAY/JUNE 2013)

Class: It is defined as blueprint or it is a collection of objects

Objects: is an instance of a class

Example:

```
class point
{
double x, y; // implicitly private
public:
void print();
void set( double u, double v );  };
```

33. When do we declare a member of a class static? (NOV/DEC 2009) (NOV/DEC 2013)

When the same data needs to be accessible by multiple instances of a class , the member of the class should be declared static.

34. What are the C++ operators that cannot be overloaded?(MAY 2007) (NOV/DEC 2012)

→ Size operator (sizeof)

→ Scope resolution operator (::)

→ Class member access operators(. , .*)

→ Conditional operator (?:)

35. What is called pass by reference?

In this method address of an object is passed, the called function works directly on the actual arguments.

36. Define constructor (NOV/DEC 2012)

A constructor is a special member function whose task is to initialize the objects of its class. It is special because its name is same as class name. The constructor is invoked whenever an object of its associated class is created. It is called constructor because it constructs the values of data members of the class

Eg: Class integer

{……public:

integer( ); //constructor

……..}

37. Define default constructor (NOV/DEC 2011)

The constructor with no arguments is called default constructor .

Eg:Class integer

{

int m,n;

public: integer( );…….};

integer::integer( )//default constructor

{ m=0;n=0; }

the statement integer a; invokes the default constructor.

38. Define parameterized constructor

Constructor with arguments is called parameterized constructor

Eg:

```
class integer
{
int m,n;
public: integer(int x,int y)
{ m=x; n=y; }
```

To invoke parameterized constructor we must pass the initial values as arguments to the constructor function when an object is declared.

This is done in two ways

1.By calling the constructor explicitly **eg:** integer int1=integer(10,10);

2.By calling the constructor implicitly **eg:** integer int1(10,10);

39. Define default argument constructor

The constructor with default arguments are called default argument constructor

Eg:

Complex (float real, float imag=0);

The default value of the argument imag is 0.

The statement complex a(6.0) assign real=6.0 and imag=0

the statement complex a(2.3,9.0)assign real=2.3 and imag=9.0

40. What is the ambiguity between default constructor and default argument constructor?

The default argument constructor can be called with either one argument or no arguments. When called with no arguments, it becomes a default constructor. When both these forms are used in a class, it cause ambiguity for a statement such as A a; The ambiguity is whether to call A::A() or A::A(int i=0)

41. Define copy constructor

A copy constructor is used to declare and initialize an object from another object. It takes a reference to an object of the same class as an argument.

Eg: integer i2(i1); would define the object i2 at the same time initialize it to the values of i1. Another form of this statement is

Eg: integer i2=i1;

The process of initializing through a copy constructor is known as copy initialization.

42. Define dynamic constructor

Allocation of memory to objects at time of their construction is known as dynamic constructor. The memory is allocated with the help of the NEW operator

Eg:

```
class string
{
char *name; int length;
public:
string( )
{
length=0;
name=new char[length +1];
}
void main( )
{
string name1("Louis"),
name3(Lagrange);
}
```

43. Define destructor

It is used to destroy the objects that have been created by constructor. Destructor name is same as class name preceded by tilde symbol (~)

Eg: ~integer(){}

A destructor never takes any arguments nor it does it return any value. The compiler upon exit from the program will invoke it. Whenever new operator is used to allocate memory in the constructor, we should use delete to free that memory.

44. Define multiple constructors (constructor overloading).

The class that has different types of constructor is called multiple constructors

Eg:

```
#include<iostream.h>
#include<conio.h>
class integer
{int m,n;
public:
integer( ) //default constructor
{m=0;n=0;}
integer(int a,int b) //parameterized constructor
{m=a; n=b;}
integer(&i) //copy constructor
{m=i.m;n=i.n;}
void main()
{
integer i1; //invokes default constructor
integeri2(45,67);//invokes parameterized constructor
integer i3(i2); //invokes copy constructor
}
```

45. Write some special characteristics / properties of constructor (NOV/DEC 2012)

- They should be declared in the public section
- They are invoked automatically when the objects are created
- They do not have return types, not even void and therefore, and they cannot return values
- They cannot be inherited, though a derived class can call the base class
- They can have default arguments

•Constructors cannot be virtual function

46. How the objects are initialized dynamically?

To call parameterized constructor we should the pass values to the object ie, for the constructor integer (int a, int b),it is invoked by integer a(10,18) this value can be get during run time. i.e., for above constructor

```
int p,q;
cin>>p>>q;
integer a(p,q);
```

47. Explain return by reference with an example.

A function can also return a reference. Consider the following function

```
int & max( int &x , int &y)
{
   if(x>y)
     return x;
  else
     return y;   }
```

Since the return type of max ( ) is int & the function returns reference to x or y (and not the values). Then a function call such as max (a , b) will yield a reference to either a or b depending on their values. The statement max (a , b) = -1; is legal and assigns –1 to a if it is larger, otherwise –1 to b.

48. What are Friend functions? Write the syntax (MAY/JUNE 2013)

A function that has access to the private member of the class but is not itself a member of the class is called friend functions. The general form is

friend data_type function_name( );

Friend function is preceded by the keyword 'friend'.

49. Write some properties of friend functions.

Friend function is not in the scope of the class to which it has been declared as friend. Hence it cannot be called using the object of that class. Usually it has object as arguments.

- It can be declared either in the public or private part of a class.
- It cannot access member names directly.
- It has to use an object name and dot membership operator with each member name. **eg:** (A.x)

50. What is function overloading? Give an example.

Function overloading means we can use the same function name to create functions that perform a variety of different tasks.

Eg: An overloaded add ( ) function handles different data types as shown below.

// Declaration is

int add(int a, int b); //add function with 2 arguments of same type int, int.

int add(int a, int b, int c); //add function with 3 arguments of same type int, int, int. double add(int p, double q); //add function with 2 arguments of different type

//Function calls

add (3 , 4); //uses prototype (i)

add (3, 4, 5); //uses prototype (ii)

add (3 , 10.0); //uses prototype (iii)

51. What is operator overloading?

C++ has the ability to provide the operators with a special meaning for a data type. This mechanism of giving such special meanings to an operator is known as Operator overloading. It provides a flexible option for the creation of new definitions for C++ operators.

The general form is

return type classname :: operator (op-arglist) { function body }

where

return type is the type of value returned by specified operation.

op - operator being overloaded. The op is preceded by a keyword operator. operator op is the function name.

52. What are the advantages of operator overloading? (NOV/DEC 2010)

- Perform different operations on the same operands
- Makes code much more readable
- Extensibility: An operator will act differently depending on the operands provided.

53. Write at least four rules for Operator overloading.

- Only the existing operators can be overloaded.
- The overloaded operator must have at least one operand that is of user defined data type.
- The basic meaning of the operator should not be changed.
- Overloaded operators follow the syntax rules of the original operators.
- They cannot be overridden.

54. How an overloaded operator can be invoked using Friend functions?

In case of unary operators, overloaded operator can be invoked as Operator op (x); In case of binary operators, overloaded operator can be invoked as Operator op (x , y)

55. List out the operators that cannot be overloaded using Friend function.

- Assignment operator =
- Function call operator ( )
- Subscripting operator [ ]
- Class member access operator →

56. What is meant by casting operator and write the general form of overloaded casting operator.

A casting operator is a function that satisfies the following conditions. It must be a class member. It must not specify a return type. It must not have any arguments.

The general form of overloaded casting operator is

```
operator type name ( )
{..... // function statements
}
```

It is also known as conversion function.

57. What are the iteration statements used in C++?

While : repeats a statement or block while its controlling expression is true.

Syntax:

```
while(condition){ //body of loop }
```

do-while : Executes its body at least once

Syntax:

```
do{ //body of loop}while(condition);
```

for : consists of three portions initialization, conditon, termination.

Syntax: for (initialization, condition, termination){ //body }

58. What is the difference between break & continue statements?

Break: We can force immediate termination of a loop, bypassing the conditional, the loop expression & any remaining code in the body of the loop. When a break statement is encountered in a loop, the loop is terminated & the program control resumes at the next statement following the loop. **Continue:** useful to force early termination. it continue running the loop, but stop processing the remainder of the code in its body for this particular iteration

59. What are the uses of break statements?

1. It terminates a statement sequence in switch statement.
2. It can be used to exit a loop.
3. It can be used as a civilized form of goto.

60. Mention some of the restrictions while using static keyword?

- They can call other static methods.
- They must only access the static data.
- They cannot refer to
  - this
  - or
  - super
  - anyway.

61. Define data members and member functions. (NOV/DEC 2011)

The attributes in the objects are known as data members because they hold the information. The functions that operate on these data are known as methods or member functions.

62. Write the properties of static member function? (NOV/DEC 2011)

- Can be called, even when a class is not instantiated
- Cannot be declared as virtual.
- Cannot access THIS pointer.
- Can access only – static member data, static member functions, data and functions outside the class.

63. When do we declare a member of a class static? (Nov/Dec 2009)

When the same data needs to be accessible by multiple instances of a class, the member of the class should be declared static.

64. What are the advantages of operator overloading?(OR) Why is it necessary to overload an operator? (Nov/Dec 2009) (Nov/Dec 2010)

Giving additional meaning to existing operators is known as Operator overloading. By operator overloading an existing operator can be made to perform different operations than the stipulated one. It doesn't change the meaning and precedence of the original operator.

Overloading is convenience for the programmer, allowing the development of operators for user-defined data types.

65. What is scope resolution operator and how can it be used for global variable? (April/May 2011) (April/May 2010)

In C, the global version of the variable cannot be accessed from within the inner block. C++ resolves this problem by introducing a new operator:: called the scope resolution operator. It is used to uncover a hidden variable.

Syntax: :: variable name;

66. What is Proxy class?

A proxy class is a class that acts on behalf of another class. There are many uses for proxy classes, but generally they are used to simplify a complex object's interface, by hiding details that are of no relevance within the object's current context. Although it may be better to declare a stripped-down version of the object, this is only practical when you have access to the riginal class implementation (in which case the full-blown class can be derived from the stripped-down class). Rather than re-invent wheels, accessing an existing class by proxy offers a more convenient workaround. Reference counting mechanisms are another example of proxy classes.

67. Write a C++ code to swap values of two variable using reference variables in function. (Nov/Dec 2014)

Program:

```
#include <iostream.h>

// function declaration
void swap(int &x, int &y);

int main ()
{
// local variable declaration:
int a = 100, b = 200;

cout << "Before swap, value of a :" << a << endl;
cout << "Before swap, value of b :" << b << endl;

/* calling a function to swap the values using variable reference.*/
swap(a, b);

cout << "After swap, value of a :" << a << endl;
```

```
cout << "After swap, value of b :" << b << endl;

return 0;
}
```

OUTPUT:

```
Before swap, value of a:  100
Before swap, value of b:  200
After swap, value of a:   200
After swap, value of b:   100
```

68. Write a C++ code to display “pen object instantiated” and “pen object destroyed” when class for pen constructor and destructor are called.(Nov/Dec 2014)

```
#include<iostream.h>
class pen
{
public:
  //constructor
  pen() {
    cout << "Inside Constructor"<<endl;
    cout << "C++ Object created"<<endl;
  }
  //Destructor
  ~pen() {
    cout << "Inside Destructor"<<endl;
    cout << "C++ Object destructed"<<endl;
  }
};
int main( )
{
  pen p1;
  pen p2;
  return 0;
}
```

69. What is a reference variable? (April/May 2015)

A reference variable is an alias, that is, another name for an already existing variable. Once a reference is initialized with a variable, either the variable name or the reference name may be used to refer to the variable

70. What is a friend function? (April/May 2015)

A friend function of a class is defined outside that class' scope but it has the right to access all private and protected members of the class. Even though the

prototypes for friend functions appear in the class definition, friends are not member functions.

71. Differentiate between constructor and destructor.

Constructor	Destructor
The name of the constructor must be same as that of the class	The destructor has the same name as that of the class prefixed by the tilde character '~'.
A constructor can have parameter list	The destructor cannot have arguments
The constructor function can be overloaded	Destructors cannot be overloaded i.e., there can be only one destructor in a class
The constructor is executed automatically	The destructor is executed automatically when the control reaches the end of class scope.

72. List down the rules for destructor.

1) The destructor has the same name as that of the class prefixed by the tilde character '~'.
2) The destructor cannot have arguments
3) It has no return type
4) Destructors cannot be overloaded i.e., there can be only one destructor in a class
5) In the absence of user defined destructor, it is generated by the compiler
6) The destructor is executed automatically when the control reaches the end of class scope.

73. List down the rules for constructor.

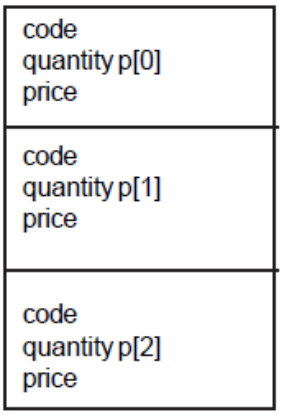
1) The name of the constructor must be same as that of the class
2) A constructor can have parameter list
3) The constructor function can be overloaded
4) The compiler generates a constructor, in the absence of a user defined constructor
5) The constructor is executed automatically

74. Write a note on array of objects?

Consider the following class definition and its corresponding memory allocation:

```
class product
{
    int code,quantity;
    float price;
public :
   void assign_Data();
   void display();
} p[3];

void main()
{
  p[0].assign_Data();
   p[0].display();
}
```



PART – B

1. State the merits and demerits of object oriented methodology. **(NOV / DEC 2007)**
2. Explain the following concepts of object oriented programming in detail with example. **(NOV / DEC 2007)**
   (i). Data abstraction
   (ii). Inheritance
   (iii). Polymorphism
   (iv). Object
3. State the rules to be followed while overloading an operator. Write a program to illustrate overloading. **(NOV / DEC 2007)**
4. (i) Describe the application of OOPs Technology. **(APRIL / MAY 2007)**
   (ii) What is an inline function? **(APRIL / MAY 2007)**
5. Illustrate the use of copy constructor and function overloading with C++ Program **(NOV / DEC 2011)**
6. Compare and contrast the following control structure with example: **(NOV / DEC 2007)**
   (i). if -- else statement & switch statement.
   (ii). Do – while and the while statement.

7.What is a friend function? What are the merits and demerits of using friend Function? **(NOV/DEC2009)**

8. Define a class 'string'. Use overload '= =' operator to compare two strings. **(NOV/DEC 2009)**
9. What is a conversion function? How is it create? Explain its syntax. **(NOV/DEC 2009)**
10. Give the syntax and usage of the reserved word in line with two examples. **(APRIL/MAY 2010)**
11. Explain the importance of constructors and destructors with example. **(APRIL/MAY 2010)**
12. Compare and contrast Structured Programming and Object Oriented Programming. **(NOV/DEC 2010)**
13. Distinguish between Data Encapsulation and Data Abstraction. **(NOV/DEC 2010)**
14. Mention the purpose of Constructor and Destructor functions. **(NOV/DEC 2010)**
15. Explain the control structures of C++ with suitable examples. **(NOV/DEC 2010)(MAY/JUNE 2013)**
16. Define function overloading with a simple example. **(NOV/DEC 2010) (MAY/JUNE 2013)**
17. What is operator overloading? Overload the numerical operators '+' and '/' for Complex numbers "addition" and "division" respectively. **(APRIL/ MAY 2011)**
18. Define friend class and specify its importance. Explain with suitable example. **(APRIL/MAY 2011)**
19. Explain the merits and demerits of object oriented paradigm. **(NOV/DEC 2011)**
20. Write a C++ program to define overloaded constructor and to perform string Initialization and string copy **(NOV/DEC 2011)**
21. Explain the concept of passing by reference using reference variables (6) **(APRIL/MAY 2011)**
22. Write a C++ program to assign 'n' projects to 'm' programmers based on the skill set of programmers using friend function. Use static variable to count total number of assignments (10) **(APRIL/MAY 2011)**
23. Define 'Copy constructor' and 'Dynamic constructor'. What are the different ways of writing copy constructor? (6) **(APRIL/MAY 2011)**
24. Where to use friend function in binary operator overloading? How? Explain with an example. (10) **(APRIL/MAY 2011)(NOV/DEC 2012)**
25. Explain the concept of "passing array of objects as an argument" with an example
26. Write a program to evaluate the equation, A = B * C using classes and objects

where A, B and C are objects of the same class (8)

27. Write a menu driven program to accept 2 integer and an operate (+,-,*, %,/) and to perform the operation and print the result.**(NOV/DEC 2012)**
28. Specify a class called complex to represent complex numbers. Overload +,-and * operators when working on the objects of this class**. (NOV/DEC 2012)**
29. Write short notes on the following
    i. Comparison of conventional programming and oops
    ii. Operator Overloading
    iii. Constructor and Destructor **(MAY/JUNE 2013)**
30. Explain with examples the types of constructors in C++ **(NOV/DEC 2013)**
31. Write a C++ Program that contains a class String and overloads the following operators on Strings.

    + To concatenate two strings

    - To delete a substring from the given string

    = = To check for the equivalence of both strings **(NOV/DEC 2013)**
32. Write a member function and friend function to subtract two complex numbers in C++.**(NOV/DEC 2014)**
33. Write a member function to perform matrix addition, simple addition and string concatenation by overloading +operator.**(NOV/DEC 2014)**
34**.** Describe the major components of Object Oriented Programming with illustrations **(April/May 2015)**
35**.** What is the purpose of constructor and destructor? Explain with suitable example the different types of constructors in C++ **(April/May 2015)**

UNIT II INHERITANCE & POLYMORPHISM

Base Classes and Derived Classes – Protected Members – Casting Class pointers and Member Functions – Overriding – Public, Protected and Private Inheritance – Constructors and Destructors in derived Classes – Implicit Derived – Class Object To Base – Class Object Conversion – Composition Vs. Inheritance – Virtual functions – This Pointer – Abstract Base Classes and Concrete Classes – Virtual Destructors – Dynamic Binding.

1. What is meant by inheritance? What is its advantage (NOV/DEC 2010) (MAY/JUNE 2013)

Inheritance is the process by which objects of one class acquire the properties of another class. It supports the concept of hierarchical classification. It provides the idea of reusability. We can add additional features to an existing class without modifying it by deriving a new class from it.

Syntax

```
class derivedClass : public baseClass {
private :
// Declarations of additional members, if needed.
public:
  // Declarations of additional members, if needed.
protected:
 // Declarations of additional members, if needed.
}
```

Uses:

- Provide the idea of reusability
- Save time and effort
- Increase program structure which results in greater reliability
- Polymorphism

2. What is special about protected inheritance?

When a base class is inherited by protected access specifier, all the private, protected and public members of the base class becomes private, protected and public member of the member functions of the derived class.

3. What does 'this pointer points to? (NOV/DEC 2009)

The keyword this identifies a special type of pointer. Suppose that you create an object named x of class A, and class A has a non-static member function f(). If you call the function x.f(), the keyword this in the body of f() stores the address of x. You cannot declare the 'this' pointer ormake assignments to it.

4. What are the representations of Big and small 'O' notations? (APRIL/MAY 2010)

1. **Big- Oh notation (O)**

   This notation is used to define the worst case running time of an algorithm and concerned with very large value of n.

   If defines the function,

   f(n) = O(g(n)).

2. **Little –oh notation**

   This notation is used to describe the worst case analysis of algorithms and concerned with small values of n.

   The function f(n)=o(g(n))

5. What in mean by binding? What is dynamic binding or late binding?(APRIL/MAY 2010)

Binding refers to the linking of a procedure to the code to be executed in response to the call. Dynamic binding means that the code associated with a given procedure call is not known until the time of the call at the run-time.

6. What are virtual functions? (NOV/DEC 2010)

A function qualified by the 'virtual' keyword is called virtual function. When a virtual function is called through a pointer, class of the object pointed to determine which function definition will be used.

7. Write some of the basic rules for virtual functions

Virtual functions must be member of some class. They cannot be static members and they are accessed by using object pointers. Virtual function in a base class must be defined. Prototypes of base class version of a virtual function and all the derived class versions must be identical. If a virtual function is defined in the base class, it need not be redefined in the derived class.

8. What are pure virtual functions? Write the syntax.

A pure virtual function is a function declared in a base class that has no definition relative to the base class. In such cases, the compiler requires each derived class to either define the function or declare it as a pure virtual function. A class containing pure virtual functions cannot be used to declare any object of its own. It is also known as "do nothing" function. The "do-nothing" function is defined as follows: virtual void display ( ) =0;

9. What is polymorphism? What are its types? (APRIL/MAY 2007)

Polymorphism is the ability to take more than one form. An operation may exhibit different behaviors in different. The behavior depends upon the type of data used. Polymorphism is of two types. They are Function overloading, Operator overloading

10. Explain basic to class type conversion with an example.

Conversion from basic data type to class type can be done in destination class. Using constructors does it. Constructor takes a single argument whose type is to be converted.

Eg: Converting int type to class type

```
class time{int hrs, mins; public:.....Time ( int t) //constructor
{
hours= t/60 ; //t in minutes mins =t % 60;} };
```

Constructor will be called automatically while creating objects so that this conversion is done automatically.

11. Explain class to basic type conversion with an example.

Using Type Casting operator, conversion from class to basic type conversion can be done. It is done in the source class itself.

Eg: vector : : operator double( ){double sum=0;

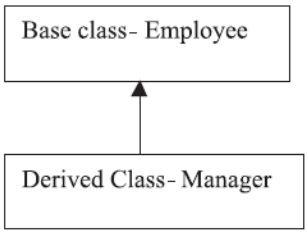
for(int I=0;I<size;I++)sum=sum+v[i] *u[i] ;return sqrt (sum) ;}This function converts a vector to the corresponding scalar magnitude.

12. Explain one class to another class conversion with an example.

Conversion from one class type to another is the combination of class to basic and basic to class type conversion. Here constructor is used in destination class and casting operator function is used in source class. Eg: objX = objY. objX is the object of class X and objY is an object of class Y. The class Y type data is converted into class X type data and the converted value is assigned to the obj X. Here class Y is the source class and class X is the destination class.

13. What is meant by single inheritance?

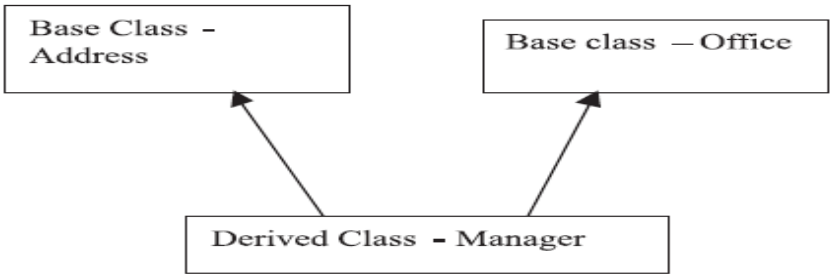
If a single class is derived from a single base class is called single inheritance.



Eg: Base class Derived class. Here class A is the base class from which the class D is derived. Class D is the public derivation of class B hence it inherits all the public members of B. But D cannot access private members of B.

14. What is multiple inheritance?(NOV/DEC 2012)

If a class is derived from more than one base class, it is called multiple inheritance. **Eg:** Base classes Derived class .Here class C is derived from two base classes A & B.

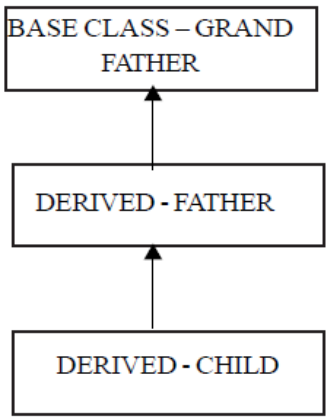


15. What is hierarchical inheritance?

If a number of classes are derived from a single base class then it is called hierarchical inheritance. Eg : Hierarchical classification of students in University ABACB

16. What is multilevel inheritance?

If a class is derived from a class, which in turn is derived from another class, is called multilevel inheritance. This process can be extended to any number of levels. **Eg:** Base class Grandfather, Intermediate, Base class Father, Derived class Child



17. What is hybrid inheritance?

It is the combination of one or more types of inheritance.

- Multilevel
- inheritance
- Multiple inheritances

The class result will have both the multilevel and multiple inheritances.

18. What is meant by Abstract base class?

A class that serves only as a base class from which derived classes are derived. No objects of an abstract base class are created. Abase class that contains pure virtual function is an abstract base class.

19. Write short notes on virtual base class.(OR) What is the need to declare base class as virtual? (NOV/DEC 2013)

A base class that is qualified as virtual in the inheritance definition. In case of multiple inheritances, if the base class is not virtual the derived class will inherit more than one copy of members of the base class. For a virtual base class only one copy of members will be inherited regardless of number of inheritance paths between base class and derived class. Eg: Processing of students' results. Assume that class sports derive the roll number from class student. Class test is derived from class Student. Class result is derived from class Test and sports as a virtual base class.

20. How the Pointer is implemented in C++? (APRIL/MAY 2011)(APRIL/MAY 2010)

Pointer is basically a variable which is used to hold the address of another variable.

Pointers can also be made available to be present in a class as follows.

Example:

```
#include <iostream.h>
#include <conio.h>
class point
{
int a,*p;
public:
void get()
{
  cout <<"\n Enter the value for a\n";
        p=&a;
        cin>>*p;
        }
        void put()
        {
        cout <<"\n The value of 'a' through pointer is\n";
        cout <<*p;
        }
           };
        void main()
        {
        clrscr();
        point p;
        p.get();
        p.put();    getch();    }
```

21. State Inheritance. (NOV/DEC 2010)

Inheritance is the most powerful feature of an object oriented programming language. It is a process of creating new classes called derived classes, from the existing or base classes. The derived class inherits all the properties of the base class. It is a power packed class, as it can add additional attributes and methods and thus enhance its functionality.

22. State Polymorphism.

Polymorphism is an important concept of OOPs. Polymorphism means one name, multiple forms. It is the ability of a function or operator to take more than one form at different instances.

23. List and define the two types of Polymorphism.

- **Operator Overloading** – The process of making an operator to exhibit different behaviors at different instances.
- **Function Overloading** – Using a single function name to perform different types of tasks. The same function name can be used to handle different number and different types of arguments.

24. Define Message Passing.

Objects communicate between each other by sending and receiving information known as messages. A message to an object is a request for execution of a procedure. Message passing involves specifying the name of the object, the name of the function and the information to be sent.

25. What effects do the visibility labels private, protected and public have on the members of a class? (NOV /DEC 2010)

Private

Specifying private visibility label is optional. By default the members will be treated as private if a visibility label is not mentioned. The members that have been declared as private can be accessed only from within the class.

Protected

The members that have been declared as protected can be accessed from within the class, and the members of the inherited classes.

Public

The members that have been declared as public can be accessed from outside the class.

26. What are the advantages of operator overloading? (NOV/DEC 2010)

Using operator overloading, we can perform different operations in different instances.

27. What is an abstract class? (NOV/DEC 2009)

Abstract class is a class that cannot be instantiated, it exists extensively for inheritance and it must be inherited.

28. When does 'this' pointer points to?(NOV/DEC 2009)

"this" pointer will point to the currently referring class object.

29. What is the use of virtual function in C++? (NOV/DEC 2013)

When the same function name is used in the base and derived classes, the function in base class is derived as virtual using the keyword virtual preceding its normal declaration.

30. Write a C++ program to print the sum of all squares between 1 and N, where N is a number accepted from the keyboard(ie) 1+4+.....+(N * N). (April/May 2008)

```
// Program to print the sum of all squares between 1 and N
#include<iostream.h>
void main( )
{
int i,n,sum=0;
cout << "Enter the number of Elements " << "\n";
cin >> n;
for (i = 0 ; i < n ; i ++)
```

```
sum = sum + i * i;
cout << “The sum of the elements is : << sum;
}
```

31. Write a C++ code to display as area of square or rectangle using function overriding.

```
#include <iostream.h>
class Rectangle
{
   protected:
     float length, breadth;
   public:
      void getdata();
};

/* Area class is derived from base class Rectangle. */
class Area : public Rectangle
{
   public:
 void getdata()
    {
     cout<<"Enter length: ";
          cin>>length;
          cout<<"Enter breadth: ";
          cin>>breadth;
       }

      float calc()
        {
           return length*breadth;
        }

     };

int main()
{
    cout<<"Enter data to find area of rectangle.\n";
    Area a;
    a.getdata();
    cout<<"Area = "<<a.calc()<<" square meter\n\n";
    return 0;
}
```

OUTPUT

```
Enter data to find area of rectangle.
Enter length: 5
Enter breadth: 4
```

Area = 20 square meter

32. Write a sample code to show the usage of this pointer in C++.

```
#include<iostream.h>
class sample
{
        int x;
   public:
        void set()
        {
                x = 27;
                cout << "\nX Value  " << this;
                this->x = 30;
                cout << "\nNew Value "<< x;
        }
};
void main()
{
        sample s;
        s.set(); }
```

Output:

```
X value  27
New Value 30
   obj.setX(x);
   obj.print();
   return 0;
}
```

OUTPUT:

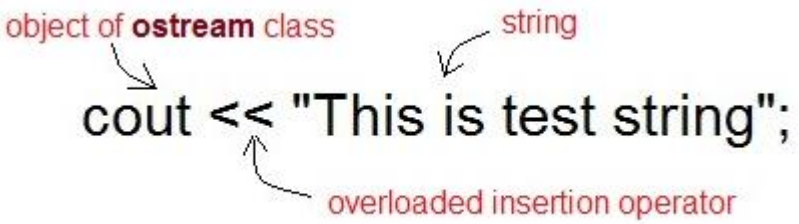
x = 20

33. What is overriding? (April/May 2015)

- Function Overloading is when multiple functions with same name exist in a class. Function Overriding is when function have same prototype in base class as well as derived class.
- Function Overloading can occur without inheritance. Function Overriding occurs when one class is inherited from another class.

34. Why there is need for operator overloading? (April/May 2015)

Overloaded operator is used to perform operation on user-defined data type. For example '+' operator can be overloaded to perform addition on various data types, like for Integer, String(concatenation) etc. Almost any operator can be overloaded in C++.



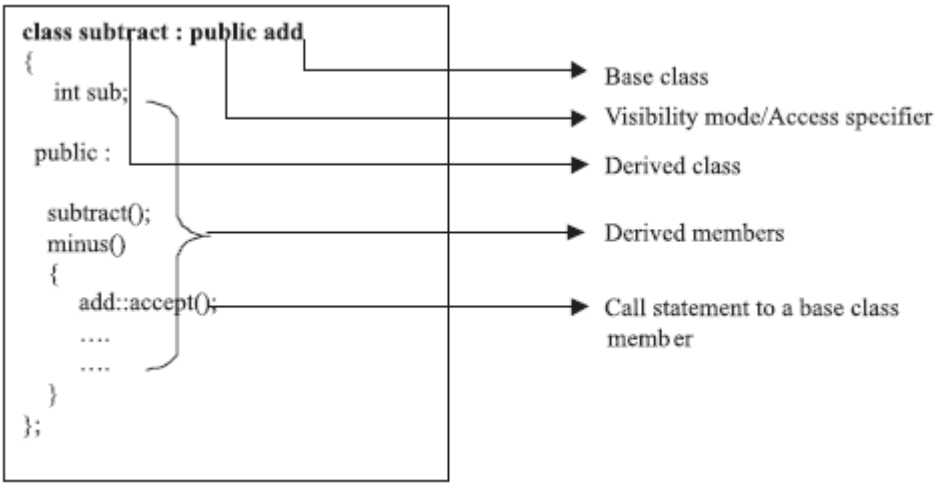
35. What are abstract classes?

Classes used only for deriving other classes are called as Abstract Classes ie., to say that objects for these classes are not declared

36. Write the syntax of derived class.

The derived class should be indicated as

class der_name : visibility mode base class-id
{
data members of the derived_class
functions members of derived_class
}



37. What are the advantages of inheritance?

1) Reusability of code : Code developed for one application can be reused in another application if such functionality is required.

2) Code sharing : The methods of the base class can be shared by the derived class.

3) Consistency of interface: The inherited attributes and methods provide a similar interface to the calling methods.

PART – B

1. Explain hybrid inheritance with suitable C++ coding. **(MAY 2007)**
2. Explain multiple inheritance with suitable c++ coding. **(NOV/DEC 2012)**
3. Define polymorphism. Explain the different types of polymorphism. **(MAY 2007)(NOV/DEC 2012)**
4. Explain multiple catch statement with help of suitable C++ coding**. (MAY 2007)**
5. Describe the various file modes and its syntax. **(DEC 2005)**
6. Discuss the need for exception with try, catch and throw keywords. **(DEC 2005)**
7. Explain the various forms of inheritance in C++ with necessary coding. **(APRIL/MAY 2008)**
8. What is a parameterized constructor? Explain with example. **(NOV/DEC 2009)**
9. Discuss Virtual function and polymorphism with example. **(APRIL/MAY 2010)**
10. Explain the concept of inheritance by considering an example of "vehicle". **(APRIL/MAY 2010)**
11. Explain the operators used for dynamic memory allocation with examples. **(APRIL/MAY 2010)**
12. Differentiate inheritance from polymorphism. **(NOV/DEC 2010)**
13. Write a C++ program to illustrate the concept of hierarchical inheritance. **(NOV/DEC 2010)**
14. Explain the exception handling mechanism of C++ in detail. **(APRIL/MAY 2011)**
15. What are the different forms of inheritance supported by C++? Explain with relevant example code? **(NOV/DEC 2011)**
16. Explain the following concepts with example code:
    i) Exception handling mechanism
    ii) Run-Time polymorphism **(NOV/DEC 2011)**
17. Write a C++ program to define a class called 'patient' (name, age, sex). Derive two classes from 'patient' namely 'in – patient' (ipno, date – of – adm, date – of – discharge) and 'out – patient' (opno, doctor – id, consultation – fee). Define two classes namely 'general – ward' (rent/day) and 'special – ward' (room_no, rent/day, eb – bill). For out – patient print the bill with consultation – fee. For in – patients, print bill according to their accommodation either in general – ward or special – ward (16) **(APRIL/MAY 2011).**

18. Write a program to maintain employee details using files. Arrange the file in descending order of their salary (8) **(APRIL/MAY 2011).**
19. Explain the concept of multiple catch statements in exception handling (5) **(APRIL/MAY 2011).**
20. Explain protected data with private and public inheritance. **(MAY/JUNE 2013).**
21. Write a C++ program to solve 8queens problem with friend function**.** **(MAY/JUNE 2013).**
22. Write an example program for virtual functions and pure virtual functions with suitable algorithm. **(MAY/JUNE 2013)**
23. Write a C++ code to construct classes of a person with name and age as public properties, account details as private properties and percentage of mark as protected property. Construct a class with sports details of person. Construct a class to rank person based on the equal weightage to academic and sports details. Use inheritance concept**. (NOV/ DEC 2014)**
24. Explain class object to base and base to class object conversion using C++ with suitable example. **(NOV/ DEC 2014)**
25. What is inheritance? Discuss in detail about the various types of inheritance in C++ with suitable example. **(April/May 2015)**
26. What is virtual function? Explain with an example how late binding is achieved using virtual function.

UNIT III LINEAR DATA STRUCTURES

Abstract Data Types (ADTs) – List ADT – array-based implementation – linked list implementation — singly linked lists –Polynomial Manipulation - Stack ADT – Queue ADT - Evaluating arithmetic expressions

1. What is an Algorithm?

An algorithm is clearly specified set of simple instructions to be followed to solve a problem. The algorithm forms a base for program.

2. What are the properties of an Algorithm?

Takes zero or more inputs Results in one or more outputs. All operations are carried out in a finite time Efficient and flexible should be concise and compact to facilitate verification of their correctness.

4. What is Complexity analysis?

It is the analysis of the amount of memory and time an algorithm requires to completion. There are two types of Complexity Space Complexity and Time Complexity.

5. Explain the performance analysis of the algorithm?

The analysis of the performance of an algorithm based on specification is called performance analysis. It is loosely divided into a. Priori estimates b. Posterior Testing.

6. Explain Space complexity?

Space complexity of an algorithm is the amount of memory it needs to run to completion.

7. Explain Time complexity?

Time complexity is the amount of computer time an algorithm requires to run to completion.

8. List out the components that are used for space complexity?

a. Instruction Space b. Environment Stack c. Data Space.

9. What do asymptotic notation means?

Asymptotic notations are terminology that is introduced to enable us to make meaningful statements about the time and space complexity of an algorithm. The different notations are Big – Oh notation Omega notation Theta notation.

10. What is meant by an abstract data type (ADT)? (MAY 2005) (MAY 2006)(NOV/DEC 2012)

An ADT is a set of operations. An ADT is a mathematical model with a collection of operations defined on that model. A useful tool for specifying the logical properties of a data type is the abstract data type. ADT refers to the basic mathematical concept that defines the data type.

11. Give few examples for data structures? (NOV/DEC 2012)

Stacks, Queue, Linked list, Trees, graphs

12. Define Efficiency of an algorithm?

It denotes the rate at which an algorithm solves a problem of size n. It is measured by the amount of resources it uses, the time and the space.

13. Define Worst case of an algorithm?

It is the longest time that an algorithm will use over all instances of size n for a given problem to produce the result.

14. Define Best case of an algorithm?

It is the shortest time that an algorithm will use over all instances of size n for a given problem to produce the result.

15. Define average case an algorithm?

It is the average time that an algorithm will use over all instances of size n for a given problem to produce the result.

16. Define Divide and Conquer algorithm?

Divide and Conquer algorithm is based on dividing the problem to be solved into several, smaller sub instances, solving them independently and then combining the sub instances solutions so as to yield a solution for the original instance.

17. Mention some application of Divide and Conquer algorithm?

a. Quick Sort

b. Merge Sort

c. Binary search

18. Define dynamic programming algorithm?

Dynamic programming algorithm is a general class of algorithms which solve problems by solving smaller versions of the problem, saving the solutions to the small problems and then combining them to solve the larger problems.

19. Mention application of dynamic programming algorithm?

- Efficient Fibonacci number computation
- Chained matrix multiplication.
- Longest common subsequence problem.

20. Define linear data structure?

Linear data structures are data structures having a linear relationship between its adjacent elements. Eg: Linked List.

21. Define Non Linear data structure?

Non Linear data structures are data structures don't have a linear relationship between its adjacent elements, but had the hierarchical relationship. **Eg:** Graph, Tree.

22. What are different types of Linked List?

- Singly linked list
- Doubly linked list
- Circular linked list

23. What are different types of Circular Linked List?

- Circular Single linked list
- Circular double linked list

24. List the basic operations carried out in a linked list?

a. Creation of list

b. Insertion of list

c. Deletion of list

d. Modification of list

e. Traversal of List

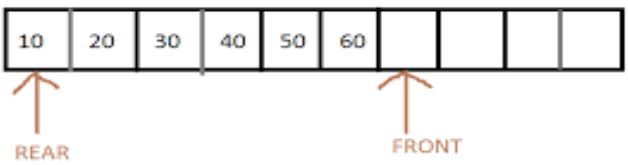
25. Define a stack? Mention the operation on stack. (MAY 2005) (MAY/JUNE 2012) (NOV/DEC2010)

Stack is an ordered collection of elements in which insertions and deletions are restricted to one end. The end from which elements are added and /or removed is referred to as top of the stack. Stacks are also referred as "piles" and "push-down lists".

Operations are PUSH and POP

26. Write short notes on queue? (April/May 2015)

Queue is an ordered collection of elements in which insertions and deletions are restricted to one end. The end from which elements are added and / or removed is referred to as the rear end and the end from which deletions are made is referred to as front end.



27. Write any 2 data structures used in operating system?(APRIL/MAY 2010)

Queue data structure is used in process scheduling

Graph data structure is used in resource allocation

28. What is single linked list?

It is a linear data structure which contains a pointer field that points to the address of its next node (successor) and the data item.

29. Define HEAD pointer and NULL pointer?

HEAD - It contains the address of the first node in that list. NULL - It indicates the end of the list structure.

30. What is meant by dummy header?

It is ahead node in the linked list before the actual data nodes.

31. Define Circular linked list?

It is a list structure in which last node points to the first node there is no null value.

32. Write operations that can be done on stack?

PUSH and POP

33. Mention applications of stack?

a. Expression Parsing

b. Evaluation of Postfix

c. Balancing parenthesis

d. Tower of Hanoi

e. Reversing a string

34. Define Infix, prefix and postfix notations?

Infix operators are placed in between the operands Prefix operators are placed before the operands Postfix Operators are placed after the operands.

35. What are the conditions that followed in the array implementation of queue?

Condition REAR <FRONTFRONT==REARREAR==ARRAY SIZE Situation EMPTY QUEUEONE ENTRY QUEUEFULL QUEUE

36. What are the conditions that could be followed in a linked list implementations of queue?

Condition REAR==HEADREAR==LINK (HEAD)NO FREE SPACE TO INSERT Situation EMPTY QUEUEONE ENTRY QUEUEFULL QUEUE

37. What is dequeue? (NOV/DEC 2009) (MAY/JUNE 2013)

Deletion or removal of data or item from a queue is known as dequeue. The meaning of dequeue is delete queue.

38. Define Circular queue?

A circular queue allows the queue to wrap around upon reaching the end of the array.

39. What are the representations of Big and small 'O' notations? (APRIL/MAY 2010)

1. Big- Oh notation (O)

This notation is used to define the worst case running time of an algorithm and concerned with very large value of n.

If defines the function,

f(n) = O(g(n)).

2. Little –oh notation

This notation is used to describe the worst case analysis of algorithms and concerned with small values of n.

The function f(n)=o(g(n))

40. What is dynamic programming (APRIL/MAY 2006) (NOV/DEC 2010)

Instead, a mathematical way of thinking about it is to look at what you should do at the end, if you get to that stage. So you think about the best

decision with the last potential partner (which you must choose) and then the last but one and so on. This way of tackling the problem backwards is Dynamic programming.

41. What is a Priority Queue? (MAY 2005)(NOV/DEC 2013)

Priority queue is a data structure in which the intrinsic ordering of the elements does determine the results of its basic operations. Ascending and descending priority queue are the two types of Priority queue.

A priority queue is a data structure that allows at least the following two operations: insert which does the obvious thing; and Deletemin, which finds, returns, and removes the minimum element in the priority queue. The Insert operation is the equivalent to enqueue.

42. What are the different ways to implement list? (MAY 2005)

- ✓ Simple array implementation of list
- ✓ Linked list implementation of list
- ✓ cursor implementation of list

43. What are the features of an efficient algorithm?

- ✓ Free of ambiguity
- ✓ Efficient in execution time
- ✓ Concise and compact
- ✓ Completeness
- ✓ Definiteness
- ✓ Finiteness

44. List down any four applications of data structures?

- ✓ Compiler design
- ✓ Operating System
- ✓ Database Management system
- ✓ Network analysis

45. What are the advantages in the array implementation of list?

- ✓ a) Print list operation can be carried out at the linear time
- ✓ b) Find Kth operation takes a constant time

46. What is a linked list?

- ✓ Linked list is a kind of series of data structures, which are not necessarily adjacent in memory. Each structure contains the element and a pointer to a record containing its successor.

47. Define max heap min heap? (DEC 2010)

- ✓ A heap in which the parent has a larger key than the child's is called a max heap.
- ✓ A heap in which the parent has a smaller key than the child is called a min heap.

48. What is a doubly linked list? (DEC 2005) (APRIL/MAY 2011)

In a simple linked list, there will be one pointer named as 'NEXT POINTER' to point the next element, where as in a doubly linked list, there will be two pointers one to point the next element and the other to point the previous element location.

49. What is meant by list ADT? (MAY 2005) (April/May 2015)

A list is a dynamic data structure. The no. of nodes on a list may vary dramatically as elements are inserted and removed. The dynamic nature of list is implemented using linked list and the static nature of list is implemented using array.

• In general the list is in the form of elements A1, A2, …, AN, where N is the size of the list associated with a set of operations:

Insert: add an element e.g. Insert(X,5)-Insert the element X after the position

Delete: remove an element e.g. Delete(X)-The element X is deleted.

Find: find the position of an element (search) e.g. Find(X)-Returns the position of X

FindKth: find the kth element e.g. Find (L,6)- Returns the element present in the given position of list L.

PrintList: Display all the elements from the list.

MakeEmpty: Make the list as empty list.

50. Application of priority queues?

1. For scheduling purpose in operating system
2. Used for external sorting
3. Important for the implementation of greedy algorithm, which operates by repeatedly finding a minimum.

51. What is a heap? Mention its types? (NOV/DEC 2009) (NOV/DEC 2010)(NOV/DEC 2012)

Heap is a binary tree, which is used to implement priority queue. Heaps is a complete binary tree or a almost complete binary tree in which every parent node be either greater or lesser than its child nodes. Binary heap satisfy heap order and structure property.

Heap must satisfy the following 2 conditions.

i. Parental dominance must hold
ii. Both the child of a parent must be present. Optionally the right may be empty.

Two types:

- Max heap
- Min heap

52. What are the main properties of a binary heap?

1. Structure property (Shape Property) 2. Heap order property

BINARY HEAPS

(A) Structure Property

– A heap is a binary tree that is completely filled, except at the bottom level, which is filled from left to right.

(**B) Heap Order Property**

– The value at any node should be smaller than all of its descendants (guarantee that the node with the minimum value is at the root).

53. Distinguish between the constraints of shape property and heap property (NOV/DEC 2011)

A **binary heap** is a heap data structure created using a binary tree. It can be seen as a binary tree with two additional constraints:

Shape property:

The tree is a complete binary tree; that is, all levels of the tree, except possibly the last one (deepest) are fully filled, and, if the last level of the tree is not complete, the nodes of that level are filled from left to right.

Heap property:

All nodes are either [greater than or equal to] *or* [less than or equal to] each of its children, according to a comparison predicate defined for the heap.

54. Write any two advantages of binary heap? (MAY /JUNE 2013)

- Quick Sort takes N^2 in worst case and NlogN average case. The worst case occurs when data is sorted. This can be mitigated by random shuffle before sorting is started.
- Quick Sort doesn't take extra memory that is taken by merge sort.
- If the dataset is large and there are identical items, complexity of Quicksort reduces by using 3 way partitions. More the no of identical items better the sort. If all items are identical, it sorts in linear time.

56. Find the maximum number of nodes in complete binary tree if d is the depth.(NOV/DEC 2014)

The maximum number of nodes in a binary tree of depth k is 2^k-1, $k\geq1$.

Here the depth of the tree. Here the depth of the tree is 1. So according to the formula, it will be $2^1-1=1$

PART – B

1. Explain how pointers are used to implement linked list structure.

2. Explain the process of inserting and deleting an element in a circular queue with an example**. (NOV/DEC 2013)**
3. Explain various operation performed on the doubly linked list. **(NOV/DEC 2008)**
4. Give linked list implementation of stack operation. **(APRIL/MAY 2011) (MAY/JUNE2013)**
5. What is a stack? Explain any two operations performed on a stack with required algorithm. **(APRIL/MAY 2008)(APRIL/MAY 2010)**
6. State and explain the priority queue with example. **(APRIL/MAY 2008) (NOV/DEC 2009) (APRIL/MAY 2010)**
7. What are the various method involved to solve hashing function?
8. Explain with example the basic heap operations and write the algorithms for the same.**(NOV/DEC 2012) (MAY/JUNE2013)**
9. Define double linked list. Explain the various operations of double linked list with algorithm. **(NOV/DEC 2009)**
10. What is hashing? Explain the various hash function with example. **(NOV/DEC 2009) (APRIL/MAY 2010)**
11. Write a C++ program to implement Stack and its operations PUSH and POP. **(APRIL/MAY 2010)**
12. Explain the operations performed on queue in detail. Write a C++ program to implement the queue operations. **(NOV/DEC 2010)(MAY/JUNE2013)**
13. Explain insertion, deletion and replacement of nodes in a heap. **(NOV/DEC 2010)**
14. What are the advantages of linked list over array.**(NOV/DEC 2010)**
15. Write an algorithm and C++ program for inserting and deleting an element from Doubly Linked List **(NOV/DEC 2011)**
16. Describe hash function. Explain the concept of Conflict Resolution Techniques in Hashing **(NOV/DEC 2011)**
17. Describe general rules to be considered while calculating time complexity of an algorithm (6) **(APRIL/MAY 2011)**
18. Write algorithms to insert an element into a stack and a queue. Can stack be used for recursion? Justify your answer. (10) **(APRIL/MAY 2011)**
19. Explain basic Heap operations and some of specialized operations with example**(APRIL/MAY 2011)**

20. Give the algorithm for inserting an element and deleting an element in a linked List **(APRIL/MAY2011)(NOV/DEC 2012)**
21. Explain with an example the formation of heap data structure and the properties to be found in a heap. **(NOV/DEC 2013)**
22. Explain the function of open-addressing and chaining in collision resolution. **(MAY/JUNE 2013)**
23. Write a C++ code to sum up all odd numbers in a single linked list. **(NOV/DEC 2015)**
24. Write a C++ code to perform addition of two polynomials using linked list form of queue.**(NOV/DEC 2015**)
25. Write a set of routines for implementing two stacks within a single array. **(April/May 2015)**
26. Write a set of routines for implementing queue using linked lists **(April/May 2015)**

UNIT IV NON-LINEAR DATA STRUCTURES

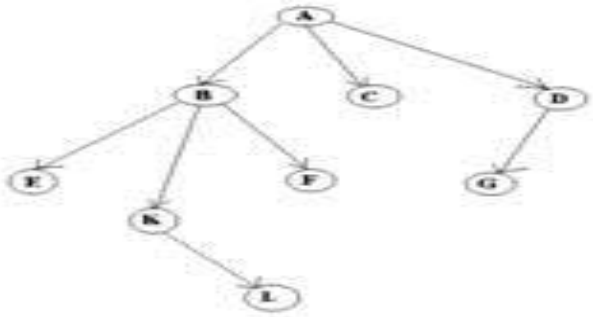
Trees – Binary Trees – Binary tree representation and traversals – Application of trees: Set representation and Union-Find operations – Graph and its representations – Graph Traversals – Representation of Graphs – Breadth-first search – Depth-first search - Connected components

1. What is a tree? (April/May 2015)

Trees are non-liner data structure, which is used to store data items in a shorted sequence. It represents any hierarchical relationship between any data Item. It is a collection of nodes, which has a distinguish node called the root and zero or more non-empty sub trees T1, T2,….Tk. each of which are connected by a directed edge from the root.

A tree can be empty with no nodes or a tree is a structure consisting of one node called the **root** and zero or one or more subtrees. A tree has following general properties:

- One node is distinguished as a **root**;
- Every node (exclude a root) is connected by a directed edge *from* exactly one other node; A direction is: *parent -> children*



A is a parent of B, C, D,
B is called a child of A.
on the other hand, B is a parent of E, F, K

2. Define Height of tree.

The height of n is the length of the longest path from root to a leaf. Thus all leaves have height zero. The height of a tree is equal to a height of a root.

3. Define Depth of tree.

For any node n, the depth of n is the length of the unique path from the root to node n. Thus for a root the depth is always zero.

4. Define Degree of a node.

It is the number of sub trees of a node in a given tree.

5. Define Degree of a tree.

It is the maximum degree of a node in a given tree.

6. Define Terminal node or leaf?

Nodes with no children are known as leaves. A leaf will always have degree zero and is also called as terminal node.

7. Define Non-terminal node?

Any node except the root node whose degree is a non-zero value is called as a non-terminal node. Non-terminal nodes are the intermediate nodes in traversing the given tree from its root node to the terminal node.

8. Define sibling?

Nodes with the same parent are called siblings.

9. Define binary tree? (NOV/DEC 2011)(NOV/DEC 2013)

A Binary tree is a finite set of data items which is either empty or consists of a single item called root and two disjoint binary trees called left sub tree max degree of any node is two.

10. What are the postfix and prefix forms of the expression? (NOV/DEC 2011)

A+B*(C-D)/(P-R)

Postfix form: ABCD-*PR-/+

Prefix form: +A/*B-CD-PR

11. Define expression tree?

Expression tree is also a binary tree in which the leafs terminal nodes or operands and non-terminal intermediate nodes are operators used for traversal.

12. Define Construction of expression trees

1. Convert the given infix expression into postfix notation
2. Create a stack and read each character of the expression and push into the stack, if operands are encountered.
3. When an operator is encountered pop 2 values from the stack.

13. Define lazy deletion?

When an element is to be deleted it is left in the tree itself and marked as being deleted. This is called as lazy deletion and is an efficient procedure if duplicate keys are present in the binary search tree, because the field that keeps count of the frequency of appearance of the element can be decremented of the element can be decremented.

14. Define AVL (NOV/DEC 2009) (NOV/DEC 2013)

AVL tree also called as height balanced tree .It is a height balanced tree in which every node will have a balancing factor of –1,0,1.Balancing factor of a node is given by the difference between the height of the left sub tree and the height of the right sub tree.

15. What are the various operation performed in the binary search tree?

1. Insertion
2. Deletion
3. Find
4. Find min
5. Find max

16. What are the various transformation performed in AVL tree?

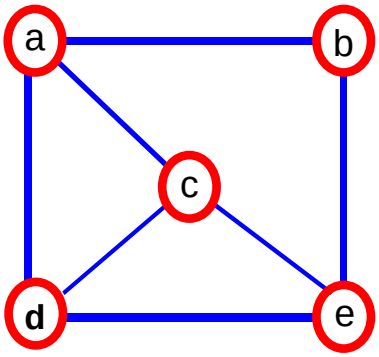
1. Single rotation- Single L rotation- Single R rotation
2. Double rotation-LR rotation-RL rotation
3.

17. How a graph is represented? (April/May 2015)

A graph consists of a set of vertices V and set of edges E which is mathematically represented as G=(V,E).Each edge in a pair(V,W) where V,W, belongs to E ,edges are sometimes referred to as arcs.

V= {a,b,c,d,e}

E= {(a,b),(a,c),(a,d),
(b,e),(c,d),(c,e),
(d,e)}



18. What are Directed graphs?

If a pair of vertices for any edge is ordered, then that graph is called as Digraph or directed graph.

19. Define Path.

A path in a graph is a sequence of vertices w1,w2w,3,wN such that Wi,Wi+1 belongs to E for a value 1<=I<=N. The length of such a path is the number of edges on the path, which is equal to n-1.

20. Define Cycle.

A cycle is a path in which the first and last vertices are the same.

21. Define Acyclic graph.

A graph with no cycles is called Acyclic graph. A directed graph with no Edges is called as a directed Acyclic graph (or)DAG. DAGS are used for Compiler Optimization process.

22. Define Connected graph.

A graph is said to be a weighted graph is connected if there is a path from every vertex to every other vertex. A directed graph with this property is called as strongly connected graph.

23. What are the conditions for a graph to become a tree?

A graph is a tree if it has two properties.

i. If it is a connected graph.

ii. There should not be any cycles in the graph.

24. Define a Weighted Graph

If every edge in the graph is assigned some weight or value. The weight of the edge is a positive value that represents the cost of moving the edge or the distance between two vertices.

25. State the properties of binary search tree. (NOV/DEC 2010)

a. The left subtree of a node contains only nodes with keys less than the node's key.
b. The right subtree of a node contains only nodes with keys greater than the node's key.
c. Both the left and right subtrees must also be binary search trees.

26. How many trees are possible with 3 nodes? (APRIL/MAY 2010)

Maximum number of trees with 'n' nodes = 2^n-n.

Maximum number of nodes in a tree = 2^3-3

=8-3

=6

27. When does a graph become a tree? (NOV /DEC 2009)

A minimum spanning tree of an undirected graph G is a tree formed graph edges that connects all the vertices of G at a lowest cost. A minimum spanning tree exists if and only if g is connected.

An acyclic graph (also known as a forest) is a graph with no cycles. A tree is a connected acyclic graph. Thus each component of a forest is tree, and any tree is a connected forest.

28. General idea of hashing and what is the use of hashing function?

A hash table similar to an array of some fixes size-containing keys. The keys specified here might be either integer or strings, the size of the table is taken as table size or the keys are mapped on to some number on the hash table from a range of 0 to table size

29. Define tree traversal and mention the type of traversals?

Visiting of each and every node in the tree exactly is called as tree traversal.

Three types of tree traversal

1. inorder traversal
2. preoder traversal
3. postorder traversal.

30. Give the types of representation of graphs.

1. Adjacency matrix
2. Adjacency linked list

31. What is a minimum spanning tree? (NOV/DEC 2012) (APRIL/MAY 2011)

A minimum spanning tree of an undirected graph G is a tree formed from graph edges that connect all the vertices of G at lowest total cost.

32. Explain about Adjacency Matrix (NOV/DEC 2010)

Adjacency matrix consists of a n*n matrix where n is the no. of vertices present. In the graph, which consists of values either 0 or 1.

33. Explain about Adjacency linked list.

It consists of a table with the no. of entries for each vertex for each entry a Linked List is initiated for the vertices adjacent to the corresponding table entry.

34. What is a single source shortest path problem?

Given as an input, a weighted graph, G = (V,E) and a distinguished vertex „S as the source vertex. Single source shortest path problem finds the shortest weighted path from s to every other vertex in G.

35. Explain about Unweighted shortest path.

Single source shortest path finds the shortest path from the source to each and every vertex present in a unweighted graph .Here no cost is associated with the edges connecting the vertices. Always unit cost is associated with each edge.

36. Explain about Weighted shortest path

Single source shortest path finds the shortest path from the source to each and every vertex present in a weighted graph. In a weighted graph some cost is always associated with the edges connecting the vertices.

37. What are the methods to solve minimum spanning tree?

a) Prims algorithm

b) Kruskal's algorithm

38. Explain briefly about Prims algorithm

Prims algorithm creates the spanning tree in various stages. At each stage, a node is picked as the root and an edge is added and thus the associated vertex along with it.

39. What is degree of graph? (NOV/DEC 2011)

The degree of the graph will be its largest vertex degree.

40. When the tree is called complete binary tree?

The tree is a complete binary tree; that is, all levels of the tree, except possibly the last one (deepest) are fully filled, and, if the last level of the tree is not complete, the nodes of that level are filled from left to right.

41. Mention the types of rotations performed on AVL trees (MAY/JUNE 2012)

- Insertion into left sub tree of left branch.
- Insertion into right sub tree of left branch.
- Insertion into left sub tree of right branch.
- Insertion into right sub tree of right branch.
-

42. When is a Binary search tree a heap? Justify (MAY/JUNE 2012)

Heap just guarantees that elements on higher levels are greater (for max-heap) or smaller (for min-heap) than elements on lower levels, whereas BST guarantees order (from "left" to "right")

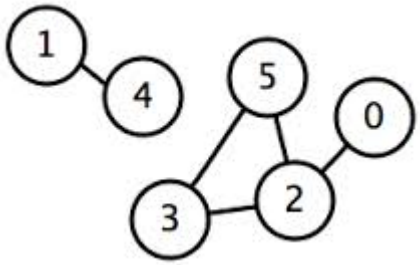
43. When the tree is called Complete Binary Tree? (NOV/DEC 2012)

- When a complete binary tree is built, its first node must be the root.
- The second node of a complete binary tree is always the left child of the root…
- The third node is always the right child of the root.
- The next nodes must always fill the next level from left to right.

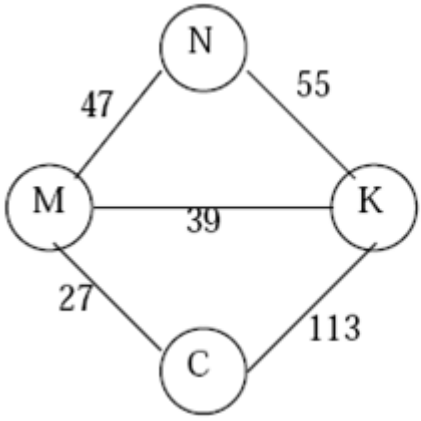
44. Write short notes on connected components. (NOV/DEC 2014)

An undirected graph is called connected if there is a path between every pair of distinct vertices in the graph. For example, any two computers in a network can communicate if and only if the graph of this network is connected. A graph consisting of only one vertex is always connected, because it does not contain any pair of distinct vertices.

Example:



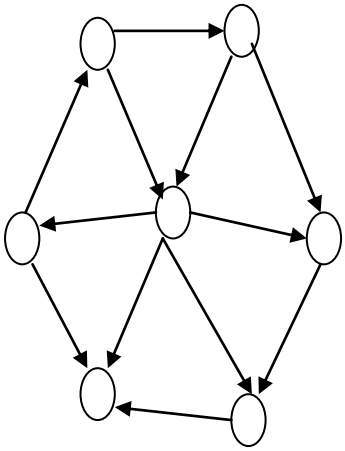
45. Give the representation of network of cities (Chennai, Delhi, Kolcutta and Mumbai) as weighted graph. (NOV/DEC 2014)



N → New Delhi
K → Kolkotta
M → Mumbai
C → Chennai

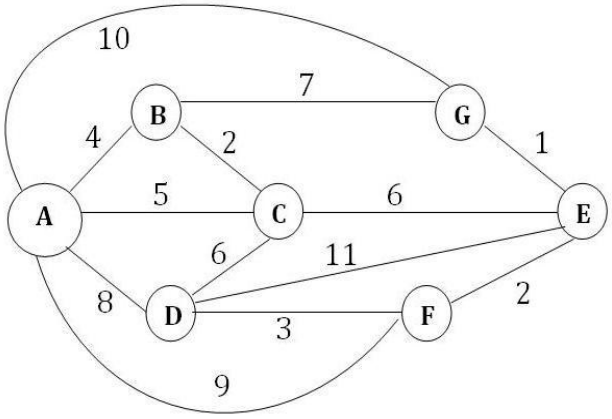
PART B

1. Explain Dijkstra's algorithm using the following graph. Find the shortest path between v1, v2, v3, v4, v5, v6 & v7. **(MAY/JUNE 2007)**



2. Write ADT operation for Prim's Algorithm. (8) **(MAY/JUNE 2007)**
3. Explain the topological sort algorithm. (8) **(MAY/JUNE 2007)**
4. Write suitable ADT operation for shortest path problem. Show the simulation of shortest path with an example graph. **(APRIL/MAY 2008)**
5. How do you construct a minimum cost spanning tree with Prim's algorithm?**(APRIL/MAY 2008)**
6. Explain depth first search on a graph with necessary data structures. (8**) (APRIL/MAY 2008)**
7. What is single source shortest path problem? Discuss Dijkstra's single source shortest path algorithm with an example. (8) **(APRIL/MAY 2007)**

8. Write an algorithm to find the minimum cost spanning tree of an undirected weighted graph. (8) **(APRIL/MAY 2007)**
9. Explain Depth – First & Breadth – First Traversal algorithms.
10. Explain Kruskal's algorithm with an example.
11. Explain Prim's algorithm with an example.
12. Discuss the different methods of traversing a binary tree with algorithm**.(NOV/DEC 2009) (APRIL/MAY 2010)**
13. Discuss prim's and kruskal's algorithm for computing the minimal spanning tree weighted undirected graph. **(NOV/DEC 2009)**
14. Convert the expression ((A+B)*C-(D-E) ^ (F+G)) to equivalent Prefix and postfix notations. **(APRIL/MAY 2011)**
15. Explain spanning tree and minimal spanning tree with examples**(NOV/DEC 2010)(NOV/DEC 13)**
16. Define AVL tree. Explain the operations on AVL tree with illustrations. **(NOV/DEC 2010)**
17. Explain breadth first search algorithm for the traversal of any graph with suitable examples. Define time complexity of the algorithm. **(NOV/DEC 2010)**
18. Describe an AVL tree. Write the algorithm to perform insertion and deletion of a node in an AVL tree and justify its worst-case with suitable example **(NOV/DEC 2011)(NOV/DEC 2013)**
19. Write the Kruskal's and Prim's algorithms for computing minimal Spanning tree **(NOV/DEC 2011)**
20. What is an AVL tree? What are the different case issues when an element is inserted into an AVL tree? How to solve those issues? **(APRIL/MAY 2011)**
21. Write down the Dijkstra's algorithm and explain it with an example (8) **(APRIL/MAY 2011)**
22. Consider the following diagram. Using Prim's algorithm and Kruskal's algorithm draw all the intermediate stages of the graph to find minimum spanning tree (8) **(APRIL/MAY 2011)**



23. Write a program to accept keys from the user one at a time, build them into an AVL tree and write out the tree at each stage. Also write a function to delete a node from AVL tree.**(MAY/JUNE 2013)**
24. Define spanning tree and minimum spanning tree. Write kruskal's algorithm for finding minimum spanning tree of any graph.**(MAY/JUNE 2013)**
25. Explain DFS and BFS with suitable example (16) **(Nov/Dec 2014)**
26. Write C++ code for the implementation of different types of tree traversals. State few tree applications.(16) **(Nov/Dec 2014)**
27. Discuss the different methods traversing a binary tree with algorithm **(April/May 2015)**
28. Illustrate the Depth First Search algorithm with a graph and explain **(April/May 2015)**

UNIT V SORTING AND SEARCHING

Sorting algorithms: Insertion sort - Quick sort - Merge sort - Searching: Linear search – Binary Search

1.What is meant by sorting? How sorting is essential for data base application? (NOV/DEC 2010)

A sorting is an algorithm that puts element of a list in a certain order. The most used orders are numerical and lexicographical order.

Sorting is essential in database applications to

- Increase the memory utilization
- Increase the stability
- Increase the execution speed

2. What is insertion sort? How many passes are required for the elements to be sorted?

One of the simplest sorting algorithms is the insertion sort. Insertion sort consist of N-1 passes. For pass P=1 through N-1 ,insertion sort ensures that the elements in positions 0 through P-1 are in sorted order .It makes use of the fact that elements in position 0 through P-1 are already known to be in sorted order.

3. Write the function in C for insertion sort?

```
void insertion_sort (int arr[], int length)
{
        int j, temp;

    for (int i = 0; i < length; i++){
        j = i;

        while (j > 0 && arr[j] < arr[j-1])
          {
            temp = arr[j];
            arr[j] = arr[j-1];
            arr[j-1] = temp;
            j--;
            }
        }
}
```

4. What are the two main classifications of sorting based on the source of data?(MAY/JUNE 2013)

a. Internal sorting

b. External sorting

- Bucket sort
- Bubble sort
- Insertion sort
- Selection sort
- Heap sort
- Merge sort

5. What is maxheap?

If we want the elements in the more typical increasing sorted order, we can change the ordering property so that the parent has a larger key than the child. It is called max heap.

6. What are the two stages for heap sort?

Stage 1: Construction of heap

Stage 2: Root deletion N-1 times

7. What is divide and conquer strategy?

In divide and conquer strategy the given problem is divided into smaller problems and solved recursively. The conquering phase consists of patching together the answers. Divide and conquer is a very powerful use of recursion that we will see many times.

8. What is the need of external sorting?

External sorting is required where the input is too large to fit into memory. So external sorting is necessary where the program is too large.

9. What is the need of external sorting?

If we have extra tapes then we can expect to reduce the number of passes required to sort our input. We do this by extending two way merge to a k-way merge.

10. Define polyphase merge? How many phases are required for k-way merge?(NOV/DEC 2011)

The k-way merging strategy requires the use of 2 k tapes. This could be prohibitive for some applications. It is possible to get by with only k+1 tapes.

11. What is replacement selection?

We read as many records as possible and sort them and writing the result to some tapes. This seems like the best approach possible until one realizes that as soon as the first record is written to an output tape the memory it used becomes available for another record. If the next record on the input tape is larger than the record we have just output then it can be included in the item. Using this we can give algorithm. This is called replacement selection.

12. What is meant by sorting? (April/May 2015)

Sorting is the process of arranging the given items in a logical order. Sorting is an example where the analysis can be precisely performed.

13. What is merge sort?

The merge sort is sorting algorithms that uses the divide and conquer strategy. In this method division is dynamically carried out.

Merge sort on an input array with n elements consists of three steps:

Divide: Partition array into two sub lists s1 and s2 with n/2 elements each.

Conquer: Then sort sub list s1 and sub list s2.

Combine: Merge s1 and s2 into a unique sorted group.

14. What are the properties involved in heap sort?

1. Structure property
2. Heap order property

15. Define articulation points.

If a graph is not biconnected, the vertices whose removal would disconnect the graph are known as articulation points.

16. What is the advantage of quick sort?

1. Quick sort reduces unnecessary swaps and moves an item to a greater distance, in one move.
2. This is faster sorting method among all.
3. Its efficiency is also relatively good.
4. It requires relatively small amount of memory

17. What is the worst case and best case no of comparisons in a linear search (NOV/DEC 2012)

The best case for a binary search is finding the target item on the first look into the data structure, so O(1). The worst case for a binary search is searching for an item which is not in the data.

In this case, each time the algorithm did not find the target, it would eliminate half the list to search through, so O(log n).

18. What is dynamic programming (APRIL/MAY 2006) (NOV/DEC 2010)

Dynamic Programming is a technique for solving problems with overlapping sub problems. The smaller sub problems are solved only once and recording the results in a table, from which the solution to the original problem is obtained.

19. What is the worst case and best case time complexity of binary tree sort? (NOVDEC 2009)

Worst case performance: O(n log n)

Best case performance: Ω(n),O(n log n)

20. What is the feature of bucket sort algorithm? (APRIL/MAY 2011)

Bucket sort, or bin sort, is a sorting algorithm that works by partitioning an array into a number of buckets. Each bucket is then sorted individually, either using a different sorting algorithm, or by recursively applying the bucket sorting algorithm. Bucket sort is a generalization of pigeonhole sort.

21. Differentiate stable and unstable sorts (MAY/JUNE 2012)

Stable sorting algorithms choose one of these, according to the following rule: if two items compare as equal, like the two 5 cards, then their relative order will be preserved, so that if one came before the other in the input, it will also come before the other in the output.

Unstable sorting algorithms can be specially implemented to be stable. One way of doing this is to artificially extend the key comparison, so that comparisons between two objects with otherwise equal keys are decided using the order of the entries in the original input list as a tie-breaker. Remembering this order, however, may require additional time and space.

22. Define spanning tree (NOV/DEC 2012)

A spanning tree T of a connected, undirected graph G is a tree composed of all the vertices and some (or perhaps all) of the edges of G. Informally, a spanning tree of G is a selection of edges of G that form a tree spanning every vertex. That is, every vertex lies in the tree, but no cycles (or loops) are formed. On the other hand, every bridge of G must belong to T.

23. What is the worst case and best case no of comparisons in a linear search (NOV/DEC 2012)

The best case for a binary search is finding the target item on the first look into the data structure, so O(1). The worst case for a binary search is searching for an item which is not in the data.

In this case, each time the algorithm did not find the target, it would eliminate half the list to search through. So O(log n).

24. What is indexed sequential search? (NOV/DEC 2009)

In indexed sequential search , the search key is compared with the index ones to find the highest index key preceding the search one, and a linear search is performed from the record the index key points onward, until the search key is matched or until the record pointed by the next index entry is reached.

25. What is the feature of bucket sort algorithm? (APRIL/MAY 2010)

Bucket sort, or bin sort, is a sorting algorithm that works by partitioning an array into a number of buckets. Each bucket is then sorted individually, either using a different sorting algorithm, or by recursively applying the bucket sorting algorithm. Bucket sort is a generalization of pigeonhole sort.

26. What is the average efficiency of heap sort?

The average efficiency of heap sort is 0 (n(log2 n)) where, n is the number of elements sorted.

27. State why quick sort is more efficient than merge sort (NOV/DEC 2013)

Quicksort is slightly sensitive to input that happens to be in the right order, in which case it can skip some swaps. Merge sort doesn't have any such optimizations. For Merge sort worst case is O(n*log(n)), for Quick sort: O(n2). Quick sort is space constant where Merge sort depends on the structure you're sorting.

28. How to perform Union operation? (NOV/DEC 2014)

There are two possible operations on set S. They are: Union & Find. The Union operation is used to add relation and find operation returns name of the set containing the element. It combines two sets into a single set.

For Example, S = ({1}, {2}, {3}, {4}, {5}).

$S1 \cap S2 = \varnothing$, $S2 \cap S3 = \varnothing$, $S1 \cap S3 = \varnothing$ and so on. It implies that the sets are disjoint.

- If we want to have an equivalence relation between two elements a and b, then find a and b.
- Then using **union** operation, we can add relationship of a with b. This creates a new set of a ~ b by destroying old disjoint sets.
- By repeatedly performing union and find operations, newer sets are created by destroying old sets. Hence this algorithm is called dynamic.

29. What is the time complexity of quick sort and binary search? (NOV/DEC 2014)

Quick Sort algorithm takes O(n log n) comparisons to sort n items. In the worst case, it makes O(n2) comparisons. Binary search algorithm takes O(log(n)) comparisons to sort n items and in the worst case it takes O(n).

30. What is time complexity? (April/May 2015)

Time complexity is the amount of computer time an algorithm requires to run to completion.

PART B

1. Write a short note on analysis of algorithm. Discuss the various notations and their complexities.
2. Explain dynamic programming with a suitable example. **(NOV/DEC 2006)**
3. Explain the various notations used in the analysis of algorithms.
4. Explain any two algorithm design techniques with suitable examples.
5. Explain divide & conquer technique applied for mergesort with a suitable example**.(MAY/JUNE 2013)**
6. Derive the best, average, worst case time complexity of a linear search **(MAY/JUNE 2007)**
7. With an example, explain how you will measure the efficiency of an algorithm. **(APRIL/MAY 2008)**
8. Write quick sort algorithm and explain.
9. State and explain the algorithm to perform heap sort. **(NOV/DEC 2010)**
10. State and explain the Huffman's algorithm.
11. Explain the all pairs shortest path algorithm with an example**. (NOV/DEC 2009)**
12. Explain the algorithm of Quick sort by sorting the following set of numbers as an example: 42 47 52 57 62 37 32 27 22 **(APRIL/MAY 2010)**
13. Describe divide and conquer technique with the help of merge sort. **(APRIL/MAY 2010) (NOV/DEC 2010)**
14. Explain insertion sort with its time complexity. **(NOV/DEC 2010) (MAY/JUNE 2013)**
15. Write the algorithms of heap sort and bucket sort. Implement the C++ program for Heap sort **(NOV/DEC 2011)**
16. Explain Greedy algorithm with an Example **(NOV/DEC 2011)**

17. What is dynamic programming? List its applications and explain any one **(NOV/DEC 2011) (NOV/DEC 2013)**
18. Give the algorithm for insertion sort (8) **(APRIL/MAY 2011)**
19. Illustrate the intermediate steps of heap tree using heap sort for the given set of numbers 67, 3, 24, 45, 9, 12, 30, 29, 90.
20. How do the Greedy algorithms work? (5) **(APRIL/MAY 2011)**
21. Write the algorithm for All – pairs shortest path and explain it with an example **(APRIL/MAY 2011)**
22. Discuss and write the program to perform topological sorting. (8)**(APRIL/MAY 2007)**
23. Sort the following values using Quick sort and estimate its time and space complexity: 70 75 80 85 60 55 50 45.Illustrate each step of the sorting process**. (NOV/DEC 2013)**
24. Compare bubble and insertion sort with an example **(MAY/JUNE 2013)**
25. Find the expected number of passes, comparisons and exchanges for shell sort when the number of elements is equal to 10.Compare this result with the actual number of operations when the given sequences is as follows: 7 , 1 , 3 , 4 , 10 , 9 , 8 , 6 , 5 , 2.**(MAY/JUNE 2013)**
26. Write C++ code to implement quick sort with suitable example. Write C++ code to implement linear search with suitable example.(16) **(NOV/DEC 2014)**
27. Write C++ code to implement merge sort with suitable example. Write C++ code to implement binary search with suitable example.(16) **(NOV/DEC 2014)**
28. Discuss the quick sort algorithm and apply the same for the following numbers 90,77,60,99,55,88,66 **(April/May 2015)**
29. Explain in detail about linear search algorithm with an example. **(April/May 2015)**