SHREE H.N.SHUKLA GROUP OF COLLEGES
(Affiliated to Saurashtra University and G.T.U)

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

SUBJECT: C++

CHAPTER-1:

Sr No.	Topic
1.	POP
2.	OOP
3.	Benefits and Application of OOP
4.	What is c++? Application and features of c++
5.	Explain input and output operators in c++
6.	Structure of c++ Program
7.	Tokens in c++
8.	Reference Variable
9.	Data Types in c++
10.	Operator
11.	Expression
12.	Assignment Operator
13.	Implicit Conversion
14.	Operator Precedence and Associativity
15.	Control Structures
16.	Loop

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

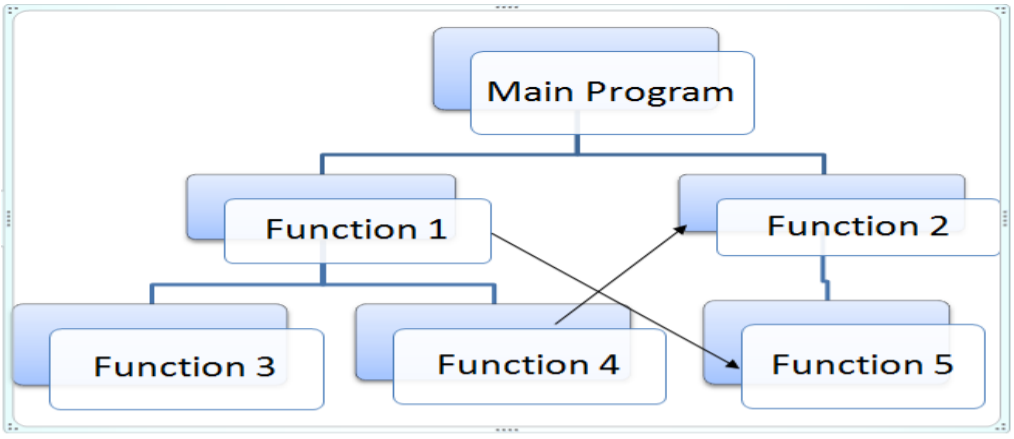
3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Topic Explain: What is procedure-oriented programming (POP)?

Ans:

** What is POP?
** Example of POP
** Difference between POP and OOP

- **Procedural programming** uses a list of instructions to tell the computer what to do step-by-step.
- It based upon the concept of the **procedure** call.
- **Procedures**, also known as routines, or functions (not to be confused with mathematical functions), but similar to those used in functional **programming**.



- **C Language is the example of POP.**
- In POP, there is no security that is one program can access information of other program that is in POP, there is no concept of access modifier.

POP	OOP
1) POP stands for procedure oriented programming	1) OOP stands for Object Oriented Programming
2) In POP, there is no security of data.	2) In OOP, data is secured.
3) POP does not have concept of access modifier	3) OOP have concept of access modifier
4) C language is example of POP	4) C++ is the example of OOP
5) In POP, the program is divided in to functions	5) In OOP, the program is divided into objects

1 word Question Answer

Sr No.	Question	Answer
1.	POP Stands for?	procedure-oriented programming

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Topic Explain: What is OOP?

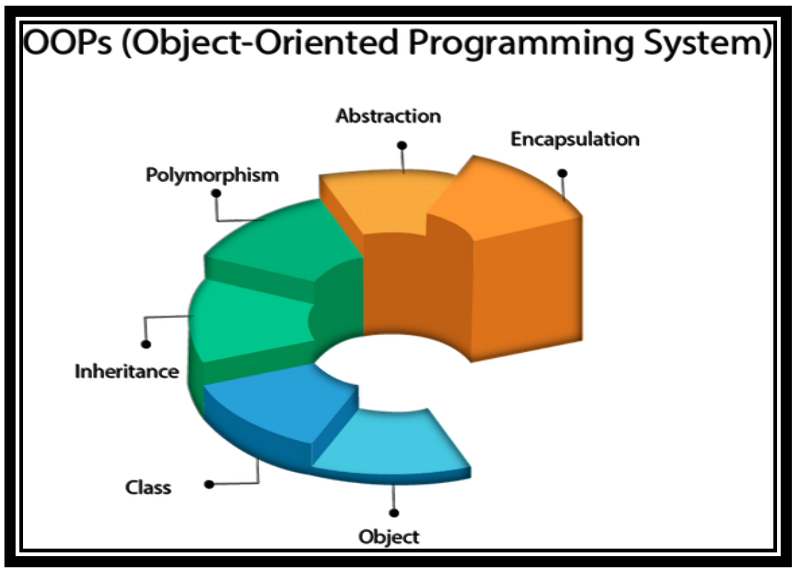
Ans:

** Full form of OOP
** What is OOP?
** Concept of class and object

CLASS
FRUIT

OBJECTS
APPLE
BANANA

- Class is the template for objects and object is the instance of the class.
- When the individual objects are created, they inherit all the variables and functions from the class.



2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Class:

- The building block of C++ that leads to Object-Oriented programming is a Class. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A class is like a blueprint for an object.
- A Class is a user-defined data-type which has data members and member functions**.**

Syntax:

```
Class A
{
    …
    …
}
```

Object:

- An Object is an identifiable entity with some characteristics and behavior.
- An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

Example:

```
Class person
{
   char name[20];
   int id;
public:
   void getdetails(){}
};

void main()
{
  person p1; // p1 is a object
}
```

1 word Question Answer

Sr No.	Question	Answer
1.	……. and …… are the components of OOP	Class and object
2.	……..is the instance of class	Object
3.	OOP stands for …….	Object Oriented Programming

Topic: Benefits and Applications of OOP

Benefits of OOP:

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs

- OOP helps to keep the C++ code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time

Applications of OOP:

- User interface design such as windows, menu.
- Real Time Systems
- Simulation and Modeling
- Object oriented databases
- AI and Expert System
- Neural Networks and parallel programming
- Decision support and office automation systems etc.

Topic: What is C++? Explain features and applications of c++

- C++ is a high level object oriented programming language that helps programmers write fast, portable programs.
- C++ is the extension of C Language.
- It was developed by BJarne Stroupstrup

Features of C++:

1) Object Oriented:
   C++ is object oriented programming language and supports all the concepts of OOP that is inheritance, Polymorphism etc.
2) Rich Library support:

Through c++ standard template library (STL) many functions are available that help in quickly writing code.

3) Speed:
   The compilation as well as execution time of c++ program is much faster than most other general purpose programming languages.
4) Pointer Support: C++ also supports pointers which are widely used in programming and are often not available in several programming languages.

Application of C++:

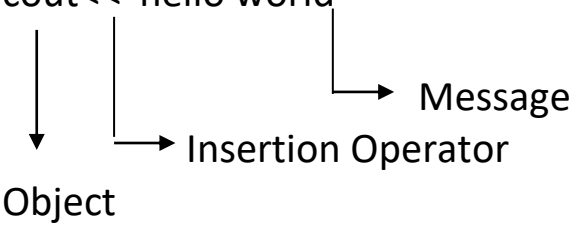
1) Operating System
2) Browsers
3) Libraries
4) Graphics
5) Banking Applications
6) Database

Topic: Explain input and output operators in c++

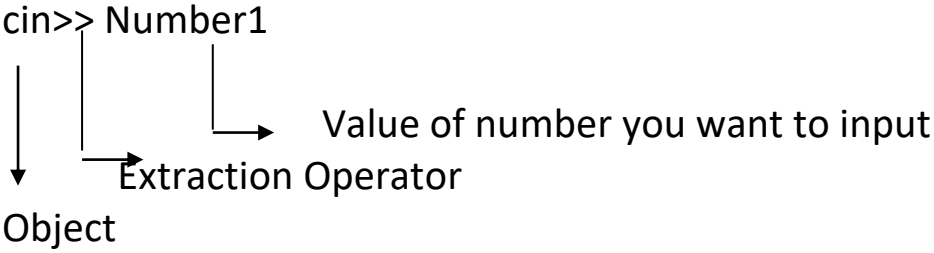
- C++ is able to input and output the built-in data types using the stream extraction operator >> and the stream insertion operator <<.
- The stream insertion and stream extraction operators also can be overloaded to perform input and output for user-defined types like an object.
- cout object is used with insertion operator. <<
- cin object is ued with extraction operator >>

Example:

cout<<"hello world"



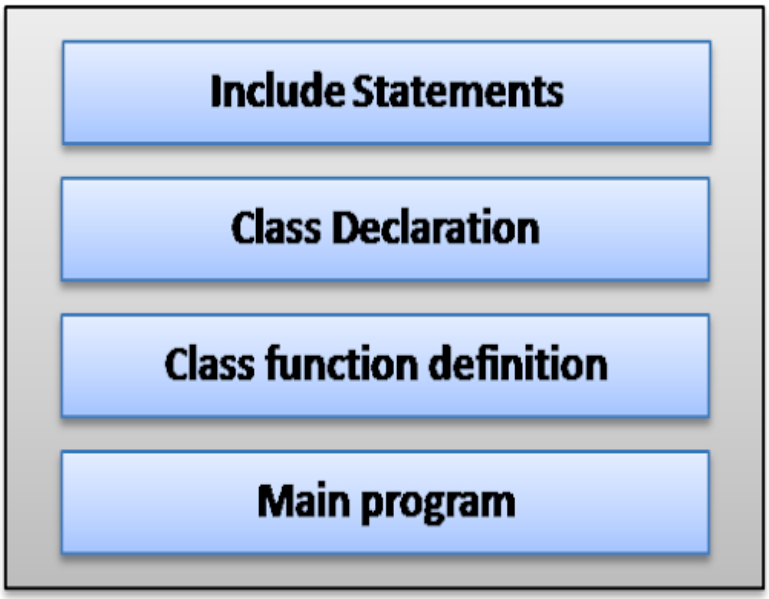
cin>> Number1



1 word Question Answer

Sr No.	Question	Answer
1.	……. operator is known as insertion operator	<<
2.	……operator is known as extraction operator	>>
3.	…….object is used with insertion operator	cout
4.	…….object is used with extraction operator	cin

Topic: Explain structure of c++ program.



1) **Include statements:**
   - Include statements are also known as pre-processor directives.
   - This statement is used to include header files in the program.
   - Example: #include<iostream.h>

2) **Class Declaration:**
   - As c++ is object oriented programming language, it uses concept of class and object.
   - So, second section that is after header file, you can create the class according to your program requirement.
   - Example:

```
Class Hello
{
        ....
        ...
};
```

Hello is name of class

3) class Function Definition/Member Function Definition:

- Class Function Definition means the body of the functions which are used inside the class.
- Example:

```
Class Hello
  {
      void print()
      {
          ...
          ...
      }
  };
```

print() is class function definition

4) Main Program:

- After the declaration of class and function, we have to declare main()
- Without main(), program does not get execute.

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Example:

```
Class Hello
  {
      void print()
      {
            …
            …
      }
  };


  void main()
  {
      Hello h;
      h.print();
      getch();
  }
```

main()

Topic: Explain tokens in c++

Ans:

- A token is the smallest element of a program that is meaningful to the compiler.
- Tokens can be classified as follows:

1. Keywords
2. Identifiers
3. Constants
4. Strings
5. Special Symbols
6. Operators

1) Keywords:

- Keywords are pre-defined or reserved words in a programming language. Each keyword is meant to perform a specific function in a program.
- Since keywords are referred names for a compiler, they can't be used as variable names because by doing so, we are trying to assign a new meaning to the keyword which is not allowed.
- In **C++,** there are **31** additional keywords other than **C** Keywords they are:

```
asm          bool       catch        class
const_cast   delete     dynamic_cast  explicit
export       false      friend       inline
mutable      namespace  new          operator
private      protected  public       reinterpret_cast
static_cast  template   this         throw
true         try        typeid       typename
using        virtual    wchar_t
```

2) Identifiers:

- Identifiers are used as the general terminology for naming of variables, functions and arrays.
- These are user defined names consisting of arbitrarily long sequence of letters and digits with either a letter or the underscore(_) as a first character
- There are certain rules that should be followed while naming C identifiers:

    1) They must begin with letter or underscore( ).
    2) They must consist of only letters, digits, or underscore, No other special character is allowed
    3) It should not be a keyword
    4) It must not contain white space
    5) It should be up to 31 characters long as only first 31 characters are significant.

Example:

```
void main()
{
        int a;
}
```

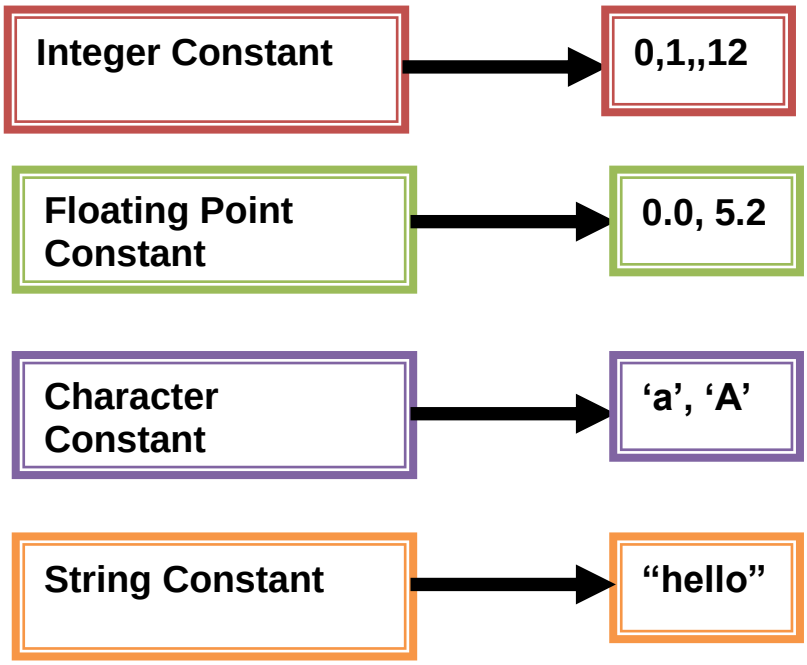
In the above example, there are 2 identifiers

1) main: Method name
2) a: variable name

3) Constant:

- Constants are also like normal variables.
- But, only difference is, their values cannot be modified by the program once they are defined.
- Constants refer to fixed values. They are also called as literals.

Types of Constant:



Note:

char string[20]="hello how are you"
char string[]="hello how are you"

- The difference in the above declaration is that when we write char string[20] then 20bytes of memory space is allocated to the string and when we write char string[] then memory space is allocated according to the requirement

4) Special Symbols:

These are the following types of symbols:

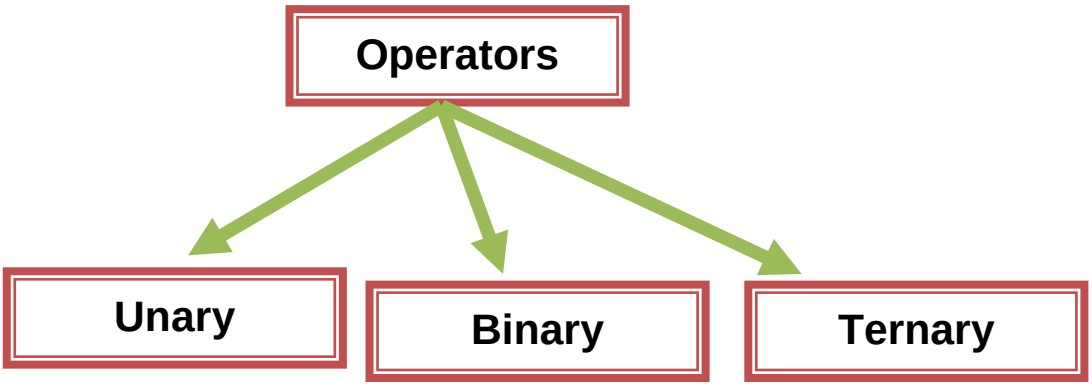
- **Brackets[]:** Opening and closing brackets are used as array element reference. These indicate single and multidimensional subscripts.
- **Parentheses():** These special symbols are used to indicate function calls and function parameters.
- **Braces{}:** These opening and ending curly braces marks the start and end of a block of code containing more than one executable statement.
- **comma (,):** It is used to separate more than one statements like for separating parameters in function calls.
- **semi colon :** It is an operator that essentially invokes something called an initialization list.
- **asterisk (*):** It is used to create pointer variable.
- **assignment operator:** It is used to assign values.
- **Pre processor(#):** The preprocessor is a macro processor that is used automatically by the compiler to transform your program before actual compilation.

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

5) Operators:

- Operators are symbols that triggers an action when applied to variables and other objects.
- The data items on which operators act upon are called operands.



Unary Operator:

- Those operators that require only single operand to act upon are known as unary operators.
- For Example increment and decrement operators

Binary Operator:

- Those operators that require two operands to act upon are called binary operators.
- Binary operators are classified into :
    1) Arithmetic operators
    2) Relational Operators
    3) Logical Operators
    4) Assignment Operators

5) Bitwise Operators

Ternary Operator:

- This operator requires three operands to act upon. For Example Conditional operator (?:).

Topic: Explain reference variable

Ans:

- A reference variable provides an alternative (alias) for previously defined variable.

Example:

```
int amount=100;
int &total =amount;
cout<<amount<<total;
```

In the above example, &total is the reference variable.

- **Features of reference variable:**
  1) A reference variable must always be initialized.
     int &j // Not valid
  2) Once a reference variable has been defined to refer to a particular variable, it cannot refer to any other variable.
  3) A variable can have multiple references.
  4) A reference of pointer can also be created.

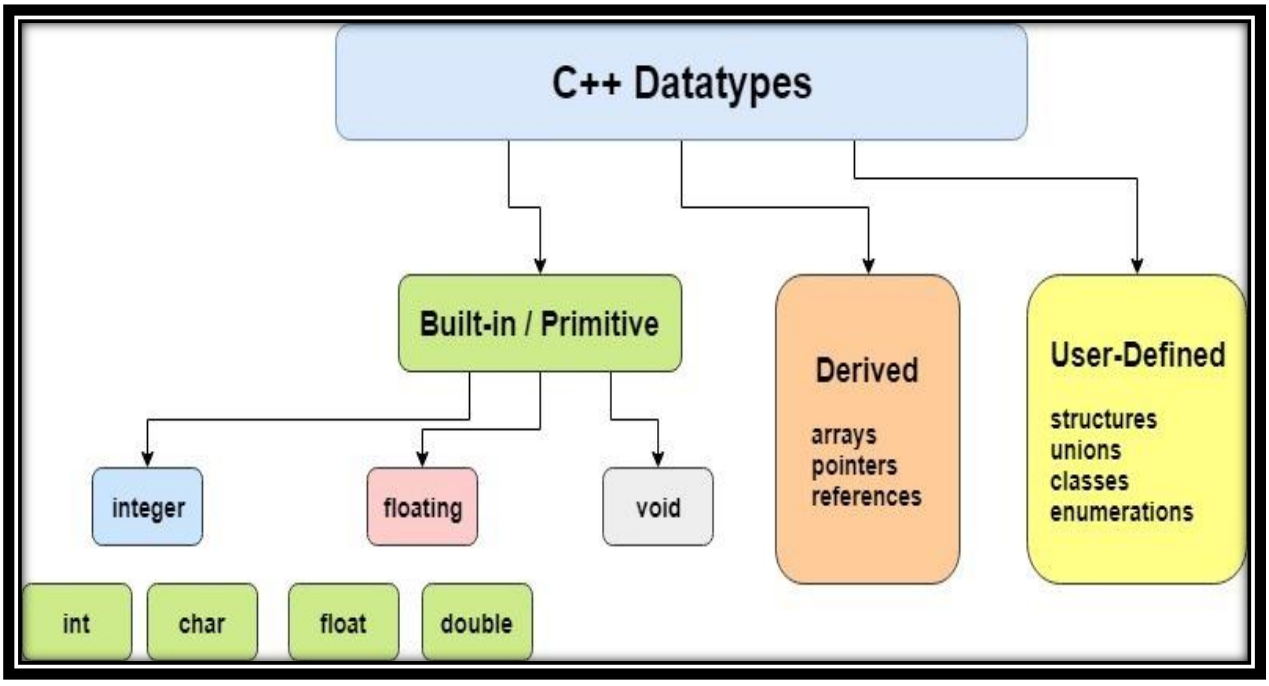
2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Topic: Explain data types in c++

Ans:

- Data types are used to tell the variables the type of data it can store.
- Whenever a variable is defined in C++, the compiler allocates some memory for that variable based on the data-type with which it is declared.



1) **Built-in data types:**

The data types which are pre-defined that is which are not created by the users is known as Built-in data types.

Example: integer, float, char etc.

2) User Defined Data Types:

The data types which are created by the user itself according to his requirements is known as user defined data types.

Example:

1) **Class:** It is new user defined data type in c++ which is the collection of variables, methods etc.

```
Class Hello
{
        void display()
        {
                Response.Write("Hello");
        }
}
```

2) **Structure:** It is the collection of variables that may or may not have same data type**.**

```
struct student
{
        int roll;
        char name[20];
};
```

3) **Union**: It is same as structure except in terms of memory location. In Union, the variable having largest data type, memory of that data type will be allocated to the entire union.

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

```
Union Student
{
        int roll;
        char name[20];
};
```

4) **Enumeration:** This data type is used to hold multiple constant values.

```
enum week={"Sunday", "Monday", ……}
```

3) **Derived Data Types:**

The data types which are derived from built in data types is known as derived data types.

Example:

1) **Function:** This is the derived data type in which user can code according to his requirement

```
void add()
{
    int a=5,b=5,sum;
    sum=a+b;
    cout<<"Sum is"<<sum;
}
```

2) **Array:** It is the collection of elements with same data types.

```
int a[5];
```

3) **Pointer:** It is also the derived data type which is used to store the address of another variable,

```
int a=10;
int *p;
p=&a;
```

Topic: Explain Operators in c++

Ans:

1) **Scope Resolution Operator(: :)**
   - This operator is used to display global variables which have same name as local variable.

Example:

```
int a=10; //Global Variable
void main()
{
        int a=10;//Local Variable
        cout<<"Local Variable<<a;
        cout<<"Global Variable"<::a;
        getch();
}
```

2) Member Dereferencing Operator:

- A **dereference operator**, also known as an **indirection operator**, operates on a pointer variable. It returns the location value, or l-value in memory pointed to by the variable's value.
- The deference operator is denoted with an asterisk (*****).

Example:

```
int x=1;
int *p;
p=&x;
cout<<x;
*p=2; //will change the value of x to 2
```

Note: As p is the pointer variable pointing to x, so if we change the value of x, then value of p will be affected and if we change the value of p, then value of x will be affected.

3) Memory Management Operator:

- C++ supports dynamic memory management using keywords new and delete.

new: This operator is used to allocate memory dynamically to pointer variable

Syntax: datatype *Pointervariable=new datatype;

Example: int *p=new int;

delete: This operator is used to free the memory that is allocated to pointer variable

Syntax: delete pointervariablename;

Example: delete p

4) Manipulators:

These are the operators that are used to format the data display. There are most commonly two manipulators:

1) endl
2) setw

endl: This manipulator is used in an output statement to enter the text or data in new line

Example:

```
cout<<"Hello"<<endl;
cout<<"Hi"
```

setw: This manipulator is used to set the width of output.

Example:

```
cout<<setw(10)<<"Hello"
```

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

5) Type Cast Operator:

This operator is used to explicitly typecast the data type. It is used to convert one data type into another data type.

Syntax: (type)expression;

Example:

```
main() {
  double a = 21.09399;
  float b = 10.20;
  int c ;

  c = (int) a;
  cout << "Line 1 - Value of (int)a is :" << c << endl ;

  c = (int) b;
  cout << "Line 2 - Value of (int)b is  :" << c << endl ;

  return 0;
}
```

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Topic: Write a short note on Expression

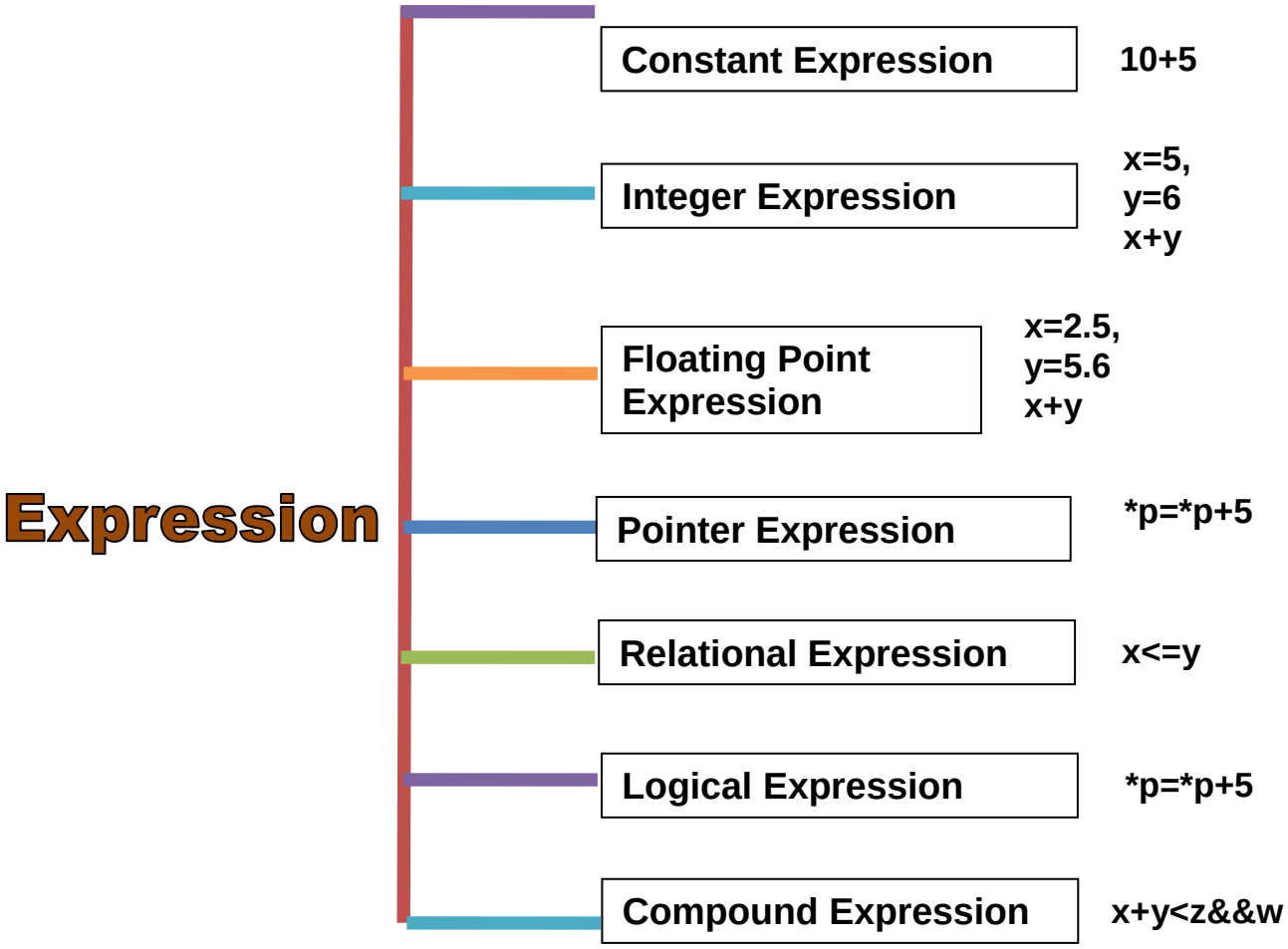
Ans:

- Expression means combination of variables, constant or symbols or operators.

Example

x+y=5;

- Following are the types of expression.



2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Topic: Write a short note on Assignment Operator

Ans:

- This operator is used to assign the value to a variable.
- = symbol is known as assignment operator.

Syntax: variable=expression

Example: a=5

- Following are the types of assignment operator.

1) **Chained Assignment:**
   - In this type of assignment, there are more than 1variables and chained assignment cannot be initialized at the time of declaration.
   - Example:

     x=y=10;

2) **Embedded Assignment:**
   - In this type of assignment, the statement is joined or embed in the assignment.
   - Example:

     x=(y=50)+10

3) Compound Assignment:

- Compound assignment is the combination of assignment operator with a binary operator.
- **It is also known as shorthand operator.**
- Example:

x=x+10 x+=10

Topic: Write a short note on Implicit Conversion

Ans:

- Whenever there are more than 1 data types in the expression, C++ performs the conversion automatically. This process is known as implicit or automatic conversion.
- Implicit conversion is also known as **silent conversion.**
- Rules for implicit conversion are:

Operand1	Operand 2	Result
char	int	int
int	long	long
int	float	float
int	double	double
double	float	double

Example:

```
int a;
float b;
```

- In the above expression, variable a has int datatype and variable b has float data type. Among that 2 data types, float is larger data type than integer so as per the rule, int is converted into float automatically.

Topic: Write a short note on Operator Precedence and associativity.

Ans:

When in the same expression, we have more than one operators then which operator should be solved first? or which operator should be given priority is the matter of consideration.

- Operator precedence means the operator having higher precedence than other operators.
- Example:

  a+b*c

- In the above example, there are 2 operators in the expression, that is + and *..but the operator * have higher precedence that + so we will solve b*c first and then result of b*c is added to a.

SHREE H.N.SHUKLA GROUP OF COLLEGES
(Affiliated to Saurashtra University and G.T.U)

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Operator	*Description*	*Associativity*
() [] . ->>> ++ --	Parentheses or function call Brackets or array subscript Dot or Member selection operator Arrow operator Postfix increment/decrement	left to right
++ -- + - ! ~ (type) * & sizeof	Prefix increment/decrement Unary plus and minus not operator and bitwise complement type cast Indirection or dereference operator Address of operator Determine size in bytes	right to left
* / %	Multiplication, division and modulus	left to right
+ -	Addition and subtraction	left to right
<< >>	Bitwise left shift and right shift	left to right
< <= > >=	relational less than/less than equal to relational greater than/greater than or equal to	left to right
== !=	Relational equal to or not equal to	left to right
&&	Bitwise AND	left to right
^	Bitwise exclusive OR	left to right
|	Bitwise inclusive OR	left to right
&&	Logical AND	left to right
||	Logical OR	left to right
? :	Ternary operator	right to left
= += -= *= /= %= &= ^= |= <<= >>=	Assignment operator Addition/subtraction assignment Multiplication/division assignment Modulus and bitwise assignment Bitwise exclusive/inclusive OR assignment	right to left
,	comma operator	left to right

- In the expression, if there are different operators which have same precedence then at that time, associativity of operators must be considered.
- Associativity of operators have 2 values:
  - left to right
  - right to left

Example:

a+b-c

In the above example, + and – both have same precedence. So in such case, associativity should be taken in to consideration. The associativity is left to right so we will solve + first and then -

Topic: Write a short note on Control Structures

Ans:

- Control statements define the direction or flow in which execution of a program should take place.
- There are 4 types of control structures:
  - if
  - if-else
  - Nested if
  - Else if Ladder
  - Switch case

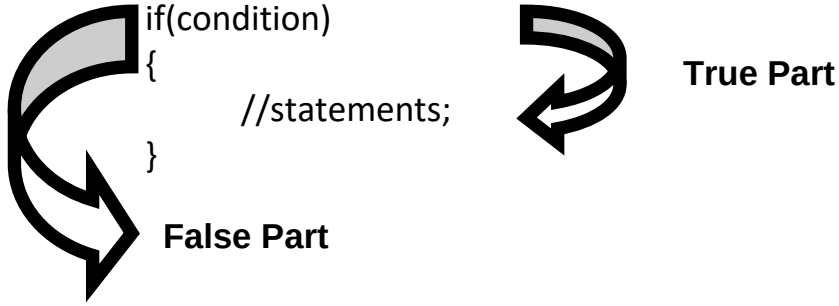
2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

1) if:

Description: This control structure is used when we have only 1 condition and only true part is taken into consideration.

Syntax:



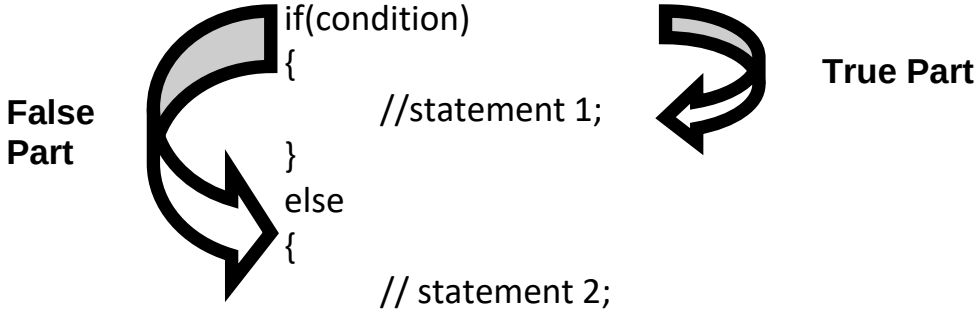
Example:

```
if(i==5)
{
        cout<<"Hello";
}
```

2) if-else:

Description: In this type of control structure, if condition is true then statement 1 is executed and if the condition is false then statement 2 is executed.

Syntax:




```
if(condition)
{
        //statement 1;
}
else
{
        // statement 2;
}
```

Example:

```
if(i==5)
{
        cout<<"Hello";
}
else
{
        cout<<"Hi";
}
```

3) Nested if:

Description:

- Nested if means one if inside another if.
- Nested if is used when you have more than 1 condition.
- In nested if, there are 2 if-else that is inner if-else and outer if-else

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Syntax:

```
if(condition 1)
{
        if(condition 2)
        {
                //statement 1;
        }
        else
        {
                //Statement 2;
        }
}
else
{
        if(condition 3)
        {
                // statement 3
        }
        else
        {
                //statement 4;
        }
}
```

True Part

False Part

Example:

```
int a=10,b=5,c=2;
if(a>b)
{
        if(a<c)
        {
```

```
                cout<<"a is max";
            }
            else
            {
                cout<<"c is max";
            }
        }
    else
    {
            if(b>c)
                cout<<"b is max";
            else
                cout<<" c is max";
    }
```

4) Else-if ladder:

Description:

- This decision control structure is also used when you work with more than 1 condition.
- if-elseif-else if-else if-else is known as else-if ladder.
- In else-if ladder when all the conditions are false then else part is executed.

Syntax:

```
if(condition 1)
{
        //Statement 1;
}
else if(condition 2)
{
        //Statement 2;
```

```
}
else if(condition 3)
{
        //Statement 3;
}
....
else
{
        //Statement 4;
}
```

Example:

```
int a=10,b=5,c=2;
if(a>b && a>c)
cout<<"a is max";
else if(b>c && b>a)
cout<<"b is max";
else
cout<<"c is max";
```

5) switch case:

Description:

- Switch case is used when you have multiple options.
- In switch case, value of variable is passed and is compared with the different cases. The value which matches the case, that case is executed.
- break keyword is used to exit from the particular case.

2 – Vaishalinagar
Nr.Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2440478

3 – Vaishalinagar
Nr. Amrapali under bridge
Raiya Road,
Rajkot – 360001.
Ph No. 0281- 2471645

Syntax:

```
switch(variable)
{
        case 1: statement 1;
                break;
        case 2: statement 2;
                break;
  default: statement 3;
           break;
}
```

Example:

```
int a;
switch(a)
{
  case 1: cout<<"Hello";
          break;
  case 2: cout<<"Hi";
          break;
  case 3: cout<<"How are you";
         break;
  default: cout<<"Invalid";
          break;
}
```

Topic: What is loop? List out looping structures and explain in detail

Ans:

- When same work is to be done multiple times, loop is used.
- There are 2 types of loop:
  - Entry controlled loop
  - Exit controlled loop

1) Entry Controlled loop:

- Entry controlled loop means the condition is checked at starting of the loop.
- There are 2 types of entry controlled loop:
  1) for loop
  2) While loop

1) for loop:

- This loop contains 3 parts: Initialization, Condition Checking, and Increment/decrement.

Syntax:

```
for(initialization; condition; increment/decrement)
{
        Statements;
}
```

Example:

```
for(i=1;i<=5;i++)
{
        cout<<"Hello";
}
```

2) **while loop:**
   - This loop also contains 3 parts: Initialization, Condition Checking, and Increment/decrement.

Syntax:

```
Initialization;
while(condition)
{
        statements;
        increment/decrement;
}
```

Example:

```
int i=1;
while(i<=5)
{
        cout<<"Hello";
        i++;
}
```

Example:

```
int i=1;
while(i<=5)
{
        cout<<"Hello";
        i++;
}
```

2) Exit Controlled loop:

- Exit controlled loop means the condition is checked at ending of the loop.
- Do-while loop is example of exit controlled loop

Syntax:

```
Initialization;
do
{
        statements;
        increment/decrement;
}while(condition);
```

Example:

```
int i=1;
do
{
        cout<<"Hello";
        i++;
}while(i<=5);
```