ABSTRACT CLASS

- An interface describes the behavior or capabilities of a C++ class without committing to a particular implementation of that class.
- The C++ interfaces are implemented using **abstract classes**
- A class is made abstract by declaring at least one of its functions as **pure virtual** function. A pure virtual function is specified by placing "= 0" in its declaration
- Animal class doesn’t have implementation of move() (assuming that all animals move), but all animals must know how to move
- We cannot create objects of abstract classes.

ABSTRACT CLASS

- *A class is abstract if it has at least one pure virtual function.*
- *We can have pointers and references of abstract class type*
- *If we do not override the pure virtual function in derived class, then derived class also becomes abstract class.*
- *An abstract class can have constructors.*

MANIPULATORS

- **Manipulators** are helping functions that can modify the input or output stream.
- It does not mean that we change the value of a variable, it only modifies the I/O stream using insertion (<<) and extraction (>>) operators.
- endl and setw are two most commonly used manipulators

MANIPULATOR

Manipulators without arguments: The most important manipulators defined by the **IOStream library** are provided below.**endl**: It is defined in ostream. It is used to enter a new line and after entering a new line it flushes (i.e. it forces all the output written on the screen or in the file) the output stream.

- **ws**: It is defined in istream and is used to ignore the whitespaces in the string sequence.
- **ends**: It is also defined in ostream and it inserts a null character into the output stream. It typically works with std::ostrstream, when the associated output buffer needs to be null-terminated to be processed as a C string.
- **flush**: It is also defined in ostream and it flushes the output stream, i.e. it forces all the output written on the screen or in the file. Without flush, the output would be the same, but may not appear in real-time.

MANIPULATOR

- **Manipulators with Arguments:** Some of the manipulators are used with the argument like setw (20), setfill (‘*’), and many more. These all are defined in the header file. If we want to use these manipulators then we must include this header file in our program.
- **Some important manipulators in *<iomanip>* are:**
  - **setw (val):** It is used to set the field width in output operations.
  - **setfill (c):** It is used to fill the character ‘c’ on output stream.
  - **setprecision (val):** It sets val as the new value for the precision of floating-point values.
  - **setbase(val):** It is used to set the numeric base value for numeric values.
  - **setiosflags(flag):** It is used to set the format flags specified by parameter mask.
  - **resetiosflags(m):** It is used to reset the format flags specified by parameter mask.

TYPE CAST OPERATOR

- C++ permits explicit type conversion of variables or expressions using type cast operator.
- (type) expression
- Example

```
float f = 34.67;
cout << int(f) << endl;
```

OPERATORS

- **Operators** are the foundation of any programming language. We can define operators as symbols that help us to perform specific mathematical and logical computations on operands. In other words, we can say that an operato
- ***Built-in operators and can be classified into 6 types:***
  1. Arithmetic Operators
  2. Relational Operators
  3. Logical Operators
  4. Bitwise Operators
  5. Assignment Operators
  6. Other Operators

ARITHMETIC OPERATORS

- These operators are used to perform arithmetic/mathematical operations on operands.
  - **Unary Operators**
    - ++val;
  - **Binary Operators**
    - cout<<a+b;

- **Relational Operators:**
  - ==, >= , <=
  - a < b
- **Logical Operators:**
  - (4 != 5) && (4 < 5);
- **Bitwise Operators:**
  - int a = 5, b = 9; // a = 5(00000101), b = 9(00001001)
  - cout << (a ^ b); // 00001100
  - cout <<(~a); // 11111010
- **Assignment Operators:**
  - “=”
  - “+=”
  - “-=”

OTHER OPERATORS

- sizeof operator
  - It is a compile-time unary operator which can be used to compute the size of its operand.
  - Basically, the sizeof the operator is used to compute the size of the variable.
- Comma operator
  - Comma as an operator
  - Comma as separator
  - Comma operator in place of a semicolon.

OPERATOR OVERLOADING

- operators with a special meaning for a data type
- Name of an operator function is always operator keyword followed by symbol of operator and operator functions are called when the corresponding operator is used
- Almost all operators can be overloaded except few.
  - . (dot)
  - ::
  - ?:
  - sizeof

EXPRESSIONS AND THEIR TYPES

- Constant Eexpression
- Integral Expressions
- Float Expressions
- Pointer Expressions
- Relational Expressions
- Logical expressions
- Bitwise Expressions

IMPLICIT TYPE CONVERSIONS

- Automatic type conversion
- Done by the compiler on its own, without any external trigger from the user.
- Generally takes place when in an expression more than one data type is present. In such condition type conversion (type promotion) takes place to avoid lose of data.
- All the data types of the variables are upgraded to the data type of the variable with largest data type.