Programming with C++

18k3MACS1

UNIT 1:

Overview of C:Introduction -Importance of C Basic Structure of C Programs - Constants,variables and Data types :Operators and Expressions: Arithmetic, Relational, Logical, Assignment, Increment, Decrement, Conditional, Bitwise, Special Operators-Arithmetic Expressions-Evaluation of Expressions -Type Conversions -Operator precedence and Associativity.

UNIT II :

Beginning with C++ - Application of C++ - More C++ Statements -Structure of C++ Program -Tokens, Expressions and Control Structures: Tokens-Keywords -Identifiers and Constants-Data Types-Declaration and Initialization of variables-Operators in C++-Expressions and their types-Operator precedence-Control Structures.

UNIT 1

CHAPTER 1

❖ OVERVIEW OF C

- History of c
- Importance of c
- Sample program
- Structure of C program

History of ANSI C

YEAR	LANGUAGE	DEVELOPER
1960	ALGOL	International group
1967	BCPL	Martin Richards
1970	B	Ken Thompson
1972	Traditional c	Dennies Ritchie
1978	K&R C	Kerningam and Ritchie
1989	ANSI C	ANSI Committee
1990	ANSI C / ISO C	ISO Committee
1999	C99	Standardization committee

Importance of C

- C is a programming Language
- C is a Robust Language
- Programs written in c are efficient and fast
- C is highly portable.i.e.., software written for one computer and run on another computer
- An important features of c is its ability to extend itself.A c program is basically a collection of functions

Format of Simple C Program

Function name	<----------	main()
start of program	<----------	{
		……
Program statements	<----------	……
		……
End of program	<----------	}

SAMPLE PROGRAM: Adding Two Numbers

```
main()
{
int  number;
float  amount;
number=100;
amount=30.75+75.35;
printf(“%d\n”,number);
printf(“%5.2f”,amount);
}
```

Output

100

106.10

Basic Structure of C Programs

Documentation section
Link section
Definition section
Global Declaration section
Main() Function section { **Declaration part** Executable part } Subprogram section (User defined fuctions) **Fun 1** Fun 2 .... Fun n

C Structure

- Documentation Section. It is the section in which you can give comments to make the program more interactive.
- Preprocessor directives Section. This section involves the use of header files that are to included necessarily
- Definition section. This section involves the variable definition and declaration in C.
- Global declaration Section. This section is used to define the global variables to be used in the programs.
- Function prototype declaration section.

CHAPTER 2

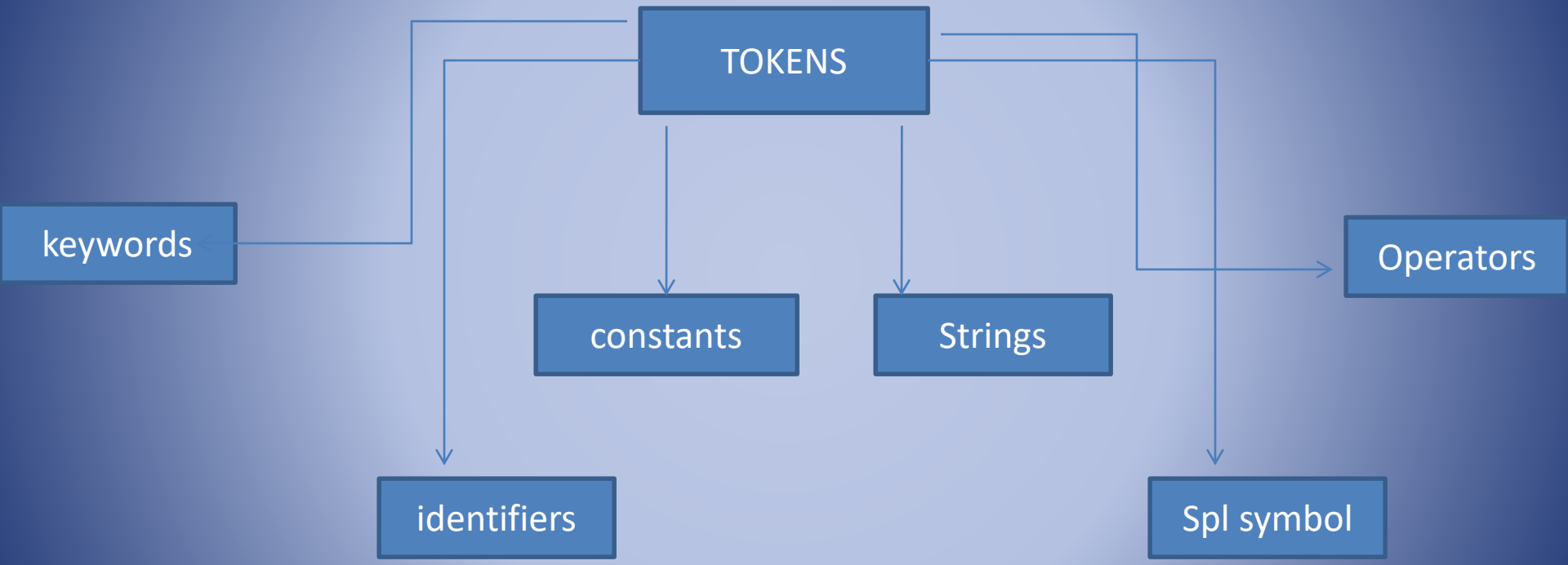
- **constants,variables and datatypes**
- Character Set
- Tokens
- Constants
- Variables
- Data types

Character set

The C character set includes the upper case letter A to Z,the lower case letter a to z ,the decimal degits 0 to 9 and certain special charecters.

- Letters — a,b,c…..z
- Digits — 0,1,2,……9
- Special Characters — , . ; : ‘? ”!|/ \~_&$%^*-+< >{}[]()#
- White Spaces — blank space,horizontal tab, carriage return,new line,form feed

TOKENS


TOKENS
keywords
constants
Strings
Operators
identifiers
Spl symbol

keywords

- Keywords are Predefined tokens in c.
- These are also called reserved words
- Keywords have special meaning to the compiler.
- These keywords can be used only for their intended action;They cannot be used for any other purpose.
- C has 32 Keywords.

C Keywords

auto	**double**	**int**	**struct**
break	else	long	Switch
case	enum	register	Typedef
char	extern	return	Union
const	float	short	Unsigned
continue	for	signed	Void
default	goto	sizeof	Volatile
do	if	static	while

Identifiers

- Identifiers are distinct names given to programs elements such as constants variables,etc…
- An identifiers is a sequence of letters,digits and the special character’_’(underscore).

1.It must start with either a letter or underscore.

2.No commas or blanks are allowed within a variable name.

3.Identifiers are case sensitive.

4.An identifiers can be of any length.

5.No special symbol can be used in a variable name.

Constants

- Constant is a literal,which remain unchanged during the execution of a program.
- Constant is a fixed value that cannot be altered during the execution of a program.
- Two types of constants.

1.Numeric constants

int,real

2.Character constants

single character,string

Integer constants

- An integer constant refer to a sequence of digits.
- It should not contain either a decimal point or exponent.
- Commas,blanks and non digit characters are not allowed in integer constants.
- The value of integer constant cannot exceed specifieds limits.The valid range is -32768 to +32767

Real constants

- Real values are often called floating point constant. There are two ways to represent a real constant decimal form and exponential form.
- In exponetial form of representation ,the real constant is represented in two parts.The part appearing before ‘e’ is called mantissa,where as the part following ‘e’ is called exponent.
  - The mantissa part and the exponential part should be separated by a letter e
  - The mantissa part may have a positive or negative sign.
  - Default sign of mantissa part is positive.
  - Exponent must have atleast one digit, which must be a positive or negative integer.default sign is positive.
  - Range of real constants expressed in exponential form is -3.4e38 to3.4e38

Character Constant

- A character constant is a single alphabet,a single digit or a single special symbol enclosed within single inverted commas. Both the inverted commas point to the left. ’A’ is valid character constant whereas A is not.
- The maximum length of a character constant can be 1 character.

  Note: every character has its ASCII value.That means every character is interchange with integer constant. ex.’A’ value is 65 and ‘a’ value is 97

String constants

- A string constant is a sequence of characters enclosed in double quotes.The character may be letters, numbers,blank space or special characters.
- Single string constant “A” is not equalent to the single character constant ‘A’.
- Each string constant must end with a special character’\0’.This character is called null character and used to terminate the string.The compiler automatically places a ‘\0’ null character at the end of every string constant

Escape sequance

- Some non-printing characters and some other characters such as double quote(“),single quote(‘),Question mark(?),and back slash(\),require an escape sequence.

A list of commanly used back slash character constant is given below:

Escape sequance	Meaning	ASCII value	Escape sequence	Meaning	ASCII value
\a	Bell	7	\r	Carriage return	13
\b	Back space	8	\”	Double quote	34
\t	tab	9	\’	Single quote	39
\n	New line	10	\?	Question mark	63
\v	Vertical tab	11	\\	back slash	92
\f	Form feed	12	\o	null	0

Variables

- A variable can be considered as a name given to the location in memory.
- The term variable is used to denote any value that is refferred to a name instead of explicit value.
- A variable is able to hold different values during execution of a program,where as constant is restricted to just one value.
- Ex..2x+3y=10;since x and y can change,They are **variables** ,whereas 2 , 3 and 10 cannot change ,hence they are **constants.**The total equation is known as **Expression.**

Rules for constructing variable name

- They must begin with a letter.some systems permit underscore as the first character.
- No special characters other than letters,digits and underscore be used in variable name.
- Commas or blanks are not allowed with a variable name.
- Uppercase and Lowercase letters are significant.That is the variable name “income”is not same as “INCOME”.Some ex of such variable names are:average,height,total,counter_1.
- Ex .of variable declarations,

```
int   average;
float  height;
```

Data Types

- A data type define set of values and the operation that can be performed on them.
- There are three classes of datatypes here:
- Primitive data types

  -int,float,double,char.
- Derived data types

  -arrays name under this category

  -arrays can contain collectin of int or float or char or double
- User dfined data types

  -structures and enum fall under this category

Size and Range of data types

Data types	Description	Size(no.of.bytes)	Range
char	Single character	1	0 to 255
Int	An integer	2	-32768 to +32767
Float	Floating point number	4	-2147483648 to +2147483647
Double	Floating point number	8	Approximately 15 digits of precision
Void	No data tyoe	0	
Signed char	Character	1	-128 to 127
Unsigned char	Unsigned character	1	0 to 255
Short signed int	Short signed integer	2	-32768 to 32767
Short unsigned int	Short unsigned integer	3	0 to 65535
Long signed int	Long signed integer	4	-2147483648 to +2147483647

CHAPTER 3

❖Operators and Expressions

- Types of Operators
- Arithmetic Expressions
- Evaluation of Expressions
- Precedence of Arithmetic Operators
- Type Conversions in Expressions
- Operator Precedence and Associativity.

Operators

- An operator is a symbol that tells the computer to perform certain mathematical or logical manipulations.
- Operators are used in program to manipulate data and variables. The data items that operators act upon are called operands.
- The operators are classified into unary, binary and ternary depending on whether they operate on one, two,or three operands respectively.

Types of Operators

- C has four classes of operators
  1. Arithmetic operators.
  2. Relational operators.
  3. Logical operators.
  4. Bit-wise operators.
- C has some special operators,which are unique to c, they are
  1. Incr and Decr Operators.
  2. Conditional Operators.
  3. Assignment Operators.

Arithmetic Operators

- There are five Arithmetic Operators in c.
- The following table lists the Arithmetic operators allowed in C:

Operators	Meaning
+	Addition
–	Subtraction ;also for unary minus
*	Multiplication
/	Division
%	Modulo Division

Relational Operators

- Relational Operators are symbol that are used to test the relationships between two variables or between a variable and constant.we often compare two quantities,and depending on their relation takes certain decisions.These comparision can be done with the help of relational operators.
- C has six relational operators as given below.

Operator	Meaning
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
==	Equal to
!=	Not equal to

Logical Operators

- Logical operators are Symbols that are used to compine or negate expressions containing relational operators.
- C has three logical operators as defined below.

Operator	Meaning
&&	Logical AND
||	LOGICAL OR
!	LOGICAL NOT

Bitwise Operators

- The lowest logical element in the memory is bit.c allows the programmer to interact directly with the hardware of a particular system through bitwise operators and expression.
- Thease operator work only with int and char datatypes and cannot be used with float and double type .
- The following table shows the bitwise operators.

Operator	Meaning
.	One’s complement
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR
<<	Shift left
>>	Shift right

Increment or Decrement Operator

- C has two very useful for adding and subtracting a variable.These are ++,--.These two operators are unary operators.
- The Increment operator ++ adds 1 to its operand and the decrement operator – subtract 1 from its operand.Therefore ,the following are equalent operators.
- ++i is equalent to i=i+1.
- --i is equalent to i=i-1.
- Thease operators are very useful in loops

Assignment Operators

- In addition to useful Assignment operator =,c has a set of short hand operators that simplifies the coding of a certain type of assignment statement.
- It is of the form

  Var op=exp ;
- Where var is a variable, op is a c binary arithmetic operator and exp is an expression.

Shorthand Assignment Operators

Statement	Equalent statement
a+=b	a=a+b
a-=1	a=a-1
a*=n+1	a=a*(n+1)
a/=n+1	a=a/(n+1)
a%=b	a=a%b

Conditional Operator

- C provides a peculiar operator ?:which is useful in redusing the code.it is ternary operator requiring three operands.
- The general format is

exp1?exp2:exp3;

Where exp1,exp2,exp3 are expressions.

- In the above conditional expression,exp1 is evaluated first.if the value of exp1 is non zero(true),then the value returned will be exp2.if the value of exp1 is zero(false),then the value returned will be exp3.

Arithmetic Expressions

- An arithmetic expression is a combination of variables,constants and operators .

Algebraic Expression	C expression
ab-c	a*b-c
(m+n)(x+y)	(m+n)*(x+y)
(ab/c)	A*b/c
3x+2x+1	3*x+2*x+1

Evaluation of Expressions

- Expressions are evaluated using an assignment statement of the form:

  variable=expression;

- Given an integer variables a,b,c,d and where a=1,b=2,c=3 and d=4.
- Evaluate the following expressions:

  x=a*b-c;

  y=b/c*a;

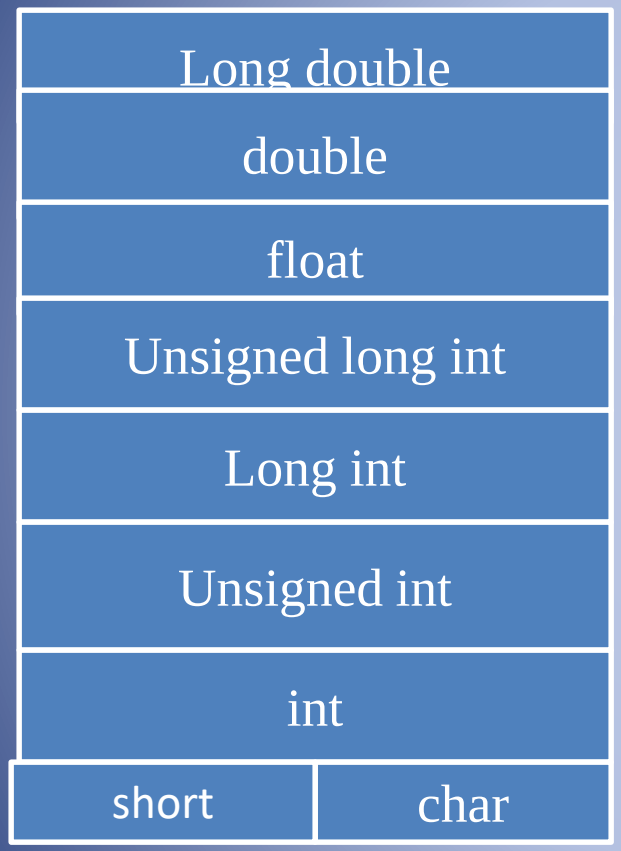
  z=a-b/c+d;

Precedence of Arithmetic operators

- The priority or precedence in which the operations of an arithmetic statement are performed is called the hierarchy (precedence) of operators.
- The operators of at the higher level of precedence are evaluated either from left to right or from right to left,depending on the level. This is known as associativity property of an operator

operator	Description	Associativity	Rank
*	Multiplication	Left to right	3
/	Division	Left to right	3
%	Modulo	Left to right	3
+	Addition	Left to right	4
-	subtraction	Left to right	4

Type conversions



1.Implicit type conversion
2.Explicit type conversion

- C performs automatic conversions of type in order to evaluate the expression. This is called implicit type conversion.
- In explicit type conversion we decide what type we want to convert the expression.
- Syntax of explicit type conversion is:
  (type) expression
- Where, **type** is any of the type we want to convert the expression into. Ex., x=(int)7.5., z=(double)sum/n.

Operator precedence and Associativity

- Operator precedence gives priorities to operators while evaluating an expression

Ex., when 2*3+2 is evaluated , output is 8 but not 12 because the * operator is having more priority than + hence 2*3 is evaluated first followed by 6+2

Operator precedence table

Operator precedence table gives the ***detail list of priorities for some operator***

- Operators are listed from higher priority to lower

Precedence	Operator	Description
1	::	Scope resolution
2	++ --	Suffix/postfix incr and decr
2	*type*() *type*{}	Function-style typecast
2	()	Function call
2	[]	Array subscripting
2	.	Element selection by reference
2	->	Element selection through pointer
3	++ --	Prefix incrand decr
3	+ -	Unary plus and minus
3	! ~	Logical NOT and bitwise NOT
3	(*type*)	C-style type cast

History of C++

- C++ evolved from C, which evolved from two previous programming languages, BCPL and B
- ANSI C established worldwide standards for C programming
- C++ "spruces up" C and provides capabilities for *object-oriented programming*
  - *objects* - reuseable software components, model things in the real world
  - Object-oriented programs are easy to understand, correct and modify

STRUCTURE OF C++ PROGRAM

HEADERS
CLASS DECLARATION
MEMBER FUNCTION DEFINITIONS
MAIN FUNCTION

SAMPLEM PROGRAM

```
#include<iostream>   \\include header file

int main()

{
        cout<< " C++ is better then C.\n ";      \\ C++ Statement
        retutn 0;
}
```

Output:

C++ is better then C

OUTPUT OPERATORS

cout << variable-name;

Meaning: print the value of variable <variable-name> to the user

cout << "any message ";

Meaning: print the message within quotes to the user

cout << endl; Meaning: print a new line

Example:

```
cout << a;
cout << b << c;
cout << "This is my character: " << my-character << " end " << endl;
```

INPUT OPERATORS

cin >> variable-name;

Meaning: read the value of the variable called <variable-name> from the user

Example:

```
cin >> a;

cin >> b >> c;

cin >> x;

cin >> my-character;
```

SAMPLE PROGRAM

```
#Include <iostream>

int main()

{
float number1,number2,sum,average;
cout<< " Enter two number:";

cin>> number1;
cin>> number2;
sum = number1 + number2;
average =  sum / 2;

cout <<" sum = " << sum <<"\n";
cout << "Average = " << average <<"\n";
return 0;
}
```

CASCADING OF I/O OPERATOR

```
cin>> number1;
cin>> number2
```

cascading Means: The Multiple use of << in one statement is called cascading

```
cin>> number1>>number2;


cout <<" sum = " << sum <<"\n";
cout << "Average = " << average <<"\n";


cout <<" sum = " << sum <<"\n" << "Average = " << average
                    <<"\n";
```

TOKENS


01
Keywords
02
Constants
03
Strings
04
Special Symbols
05
Character Set
06
Operators
Tokens in C++

Symbolic Constants

There are two ways of creating symbolic constantsin C++

- Using the qualifier const
- Defining a set of interger constants using enum keyword

C and C++ any value declared as const connot be modified by the program during the execution.

Example:

```
const  int  pi = 3.1415;
const  name [size];
enum{ X,Y,Z};
```

- This define X,Yand Z as integer constant with values 0,1,2
- We can assign values to X,Y,Z Explicitly
  - Enum{X=100,Y=50,Z=200};

Dynamic Initialization of variable

C++ Permits initialization of the variables at run time this is referred to as ***dynamic initialization***

float area = 3.14159 * rad * rad;

- ***In C++ a variable can be initialized at run time using*** *expressions* ***at the place of declaration.***
- ***Both the declaration and the initialization of avariable can be done simultaneously at the place where the variable used in the first time .***
  - ***float average;***
  - ***average = sum/i;***
- ***Combine both :*** ***float average = sum/i;***

Reference Variable

- A reference varaible provides an *alias* (altetnative name) from a previously.

  The variable sum as a reference to the variable total.

  ```
  data_type & reference-name = variable
  ```

  Example : float total =100;

  float & sum=total;

  A reference variable must be initialized at the time of declaration.

Operators in C++

- :: Scope resolution operator
- ::* Pointer-to-member declaration
- ->* Pointer-to-member operator
- .* Pointer-to-member operator
- delete Memory release operstor
- endl Line feed operator
- new Memory allocation operator
- setw Field width operator.

MANIPULATORS

C++ Manipulators:

Manipulators are operators used in C++ for formatting output. The data is manipulated by the programmer's choice of display.

In this C++ tutorial, you will learn what a manipulator is,

- endl manipulator,
- setw manipulator,
- setfill manipulator and
- setprecision manipulator

are all explained along with syntax and examples.

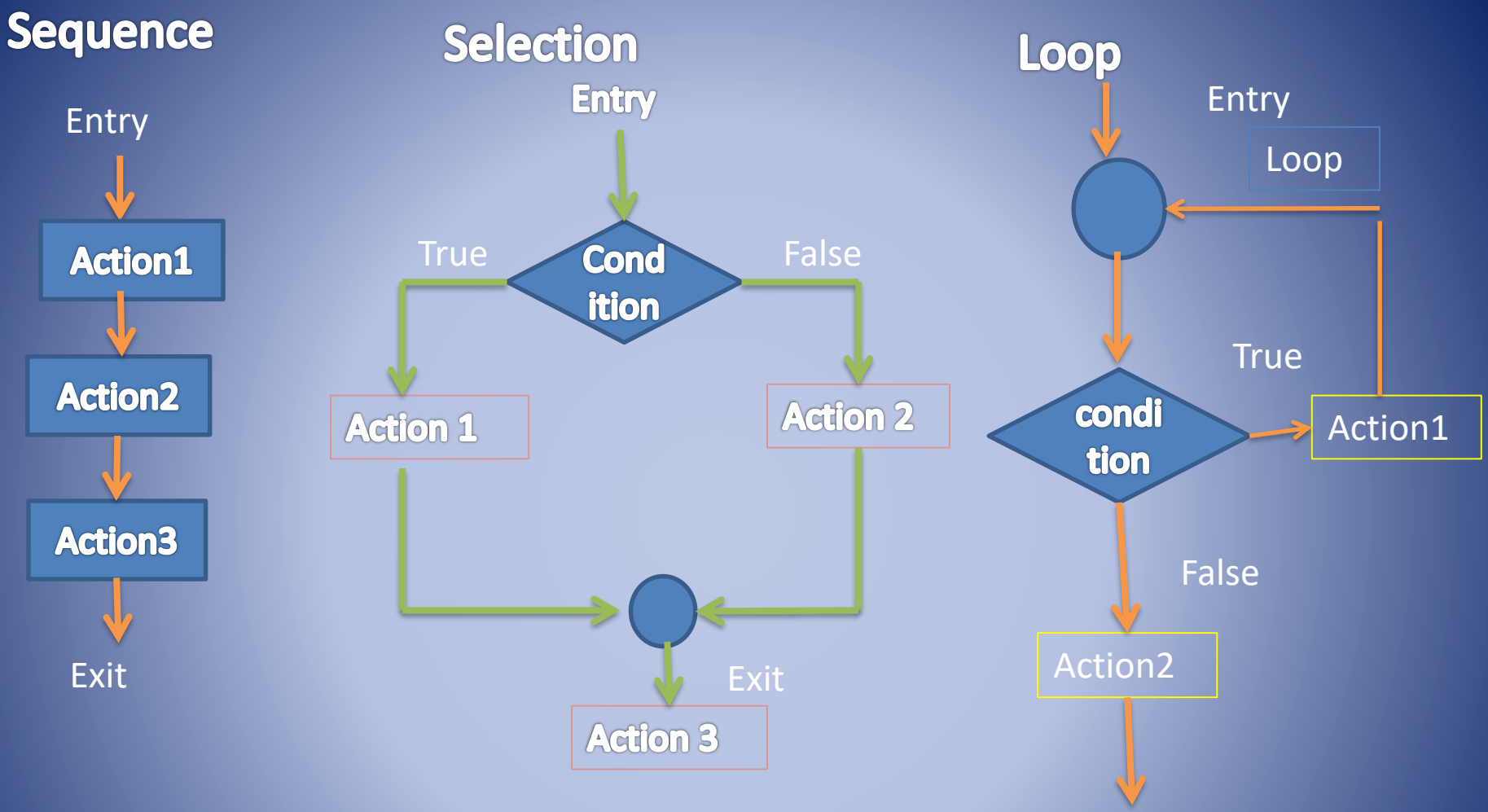
endl : linefeed to be inserted (“ \n ”)

cout<<“m= “m<<endl;

Setw :Specify the field of width(right justification)

Control Structure

- Control structure from the basic entities of “structured programming language.
- Control structures are used to alter the flow of execution of the program.
- “***decision making***“ We use control structures to make decisions and alter the direction of program flow in one or the other path(s) available.
- There are three types of control structures available in C and C++
  - ❖1) Sequence structure (straight line paths)
  - ❖2) Selection structure (one or many branches)
  - ❖3) Loop structure (repetition of a set of activities)


Sequence
Entry
Action1
Action2
Action3
Exit
Selection
Entry
True
Cond
ition
False
Action 1
Action 2
Exit
Action 3
Loop
Entry
Loop
True
condi
tion
Action1
False
Action2

CLASS : II B.Sc.,Maths [SF & SSS].
SUBJECT : PROGRAMMING WITH C++.
SUBJECT CODE : 18K3MACS1

UNIT 1: 1. Class Incharge
N.Anuradha.,M.sc.,M.Phil.,
Guest Lecturer.,
Dept of Comp.Science.,
K.N.G.A.C.,Thanjavur.

UNIT II: 2. Class Incharge
G.MuthamizhSelvi.,M.Sc.,M.Phil.,P.hd.,
Guest Lecturer.,
Dept of Comp.Science.,
K.N.G.A.C.,Thanjavur.