GOVERNMENT ARTS COLLEGE (AUTONOMOUS)

DEPARTMENT OF COMMERCE (CA)

CLASS: I M.COM (CA) **SEMESTER II**

SUBJECT CODE: 18MCC23C

OBJECT ORIENTED PROGRAMMING WITH C++

UNIT 2

Data types – character set – Tokens, Identifiers and Keywords – variables – operator and expressions – control flow – IF, IF….else, nested IF….else, For loop, While…. loop, do…while loop, break statements, switch statements continue statement and go to statement. Arrays – operations on arrays – multidimensional arrays – strings – string manipulations. Functions – function- components – library functions – inline functions.

Reference Books:

E. Balagurusamy, "Object Oriented Programming with C++", TataMcGraw Hill Publishing Company Ltd.

K.R.Venugopal, Raj kumar, T.Ravishanker., "Mastering C++",TataMcGraw Hill Publishing Company Ltd.

Prepared by

Dr. S. Vasantha

UNIT 2

Data types – character set – Tokens, Identifiers and Keywords – variables – operator and expressions – control flow – IF, IF….else, nested IF….else, For loop, While…. loop, do…while loop, break statements, switch statements continue statement and go to statement. Arrays – operations on arrays – multidimensional arrays – strings – string manipulations. Functions – function- components – library functions – inline functions.

TOKENS

- ✓ Key words
- ✓ Identifiers
- ✓ constants
- ✓ Strings
- ✓ Operators

KEY WORDS

They are explicitly reserved identifiers and cannot be used as names for the program variables or other user defined program elements.

EXAMPLE

Auto, break, case, catch, char, const, class, continue, default, delete, do, double, else, float, for, friend, go to, if, inline int, long, new, operator, private, public, return, short, signed, size of, static, struct, switch etc.

IDENTIFIERS AND CONSTANTS

Identifiers refer to the names of variables, functions, classes, etc, created by the programmer.

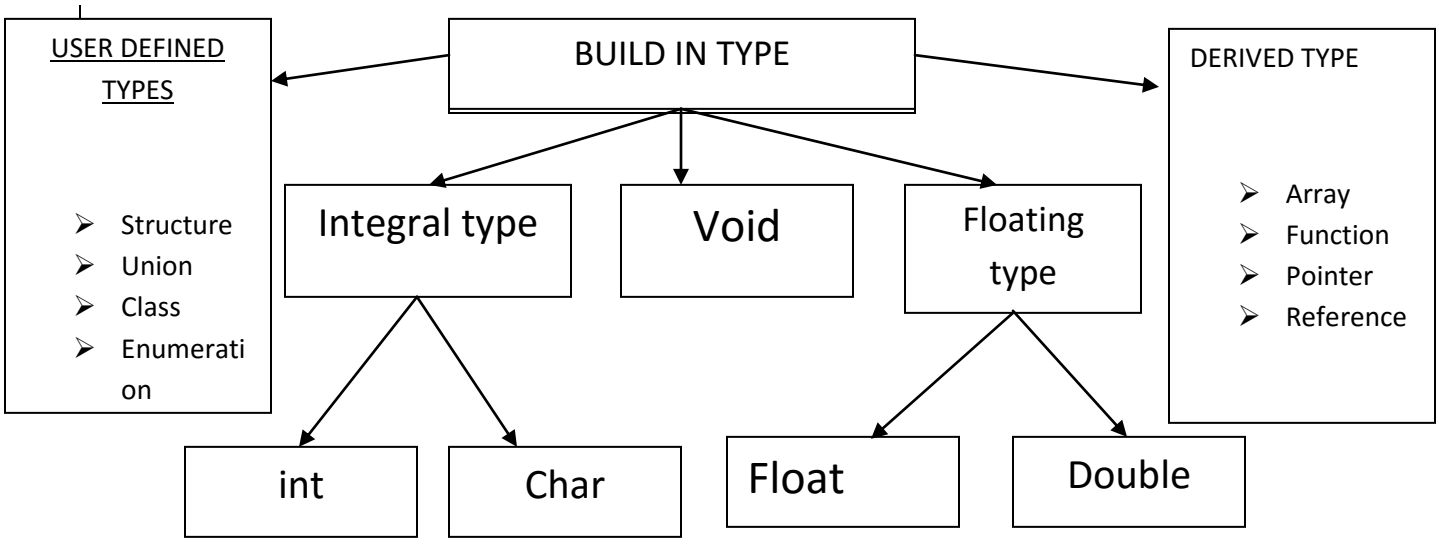
They are the fundamental requirement of any language.

RULES FOR NAMING THESE IDENTIIERS (VARIABLES):

- Only alphabetic characters, digits and underscores are permitted.
- The name cannot start with a digit.

- Uppercase and lowercase letters are district.
- A declared key word cannot be used as a variable name.
- The maximum number of characters used in forming an identifier must not exceed 32 characters.

DATA TYPES



VARIABLES

A variable is the symbolic address of a location in memory where data can be stored. Variables are object of the program elements that may change it contents of the value during program execution.

SIZE OF DATA

TYPE	BYTES	TYPE	BYTES
Char	1	Signed short int	2
Unsigned char	1	Long int	4
Signed char	1	Signed long int	4
Int	2	Unsigned long int	4
Unsigned int	2	Float	4
Signed int	2	Double	8
Unsigned short int	2	Long double	10

OPERATORS:

- Arithmetic operators
- Relational operators
- Logical operators
- Assignment operators
- Increment and decrement operators
- Conditional operators
- Bitwise operators
- Special operators in C++

ARTHIMETIC OPERATORS:

C++has both unary and binary arithmetic operators.

Unary operators are those, which operate on a single operand whereas, binary operators operate on two operands.

UNARY MINUS OPERATOR (NAGATIVE):

The unary minus operator can be used to negate the value of variable. It is also used to specify a negative number.

(e.g) int x=5; y=x; y values = -5.

Here a minus (-) sign is preferred to the umbers. The use of unary + operator does not save any purpose.

A= =100

By default, numeric constants are assumed to be positive.

BINARY OPERATORS:

Binary arithmetic operators such as +,-,*,/ and% require two operands of standard data types, depending on the data types of the operands, these operators perform either integer or floating point arithmetic operation.

An arithmetic expression without parentheses will evaluate from left to right using the following rules of precedence for operators.

High priority: *, /, %

Low priority: +, -

When parentheses are used, the expression within the innermost parentheses gains highest priority.

RELATIONAL OPERATORS:

A relational operator is used to make comparisons between two expressions. All these operations are binary and require two operands.

Logically similar quantities are often compared for taking decisions. These comparisons can be done with the help of relational operators.

Each one of these operators compares its left hand side operand with its right hand side open and the whole expression involving the relational operator then evaluates to an integer. It evaluates to zero if the condition is false and non zero value if it is true.

OPERATOR	MEANING
<	Less then
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not equal to

EXAMBLE:

```
# include <iostream.h>

void main ()

{

    Int my_age, your_age;
```

```
Cout<< “enter my_age:”;

Cin>> my_age;

Cout<< “Enter your_age:”;

Cin your_age;

If( my _age==your_age)

Cout<< “We are born in the same year”;

Else

Cout<< “ We are born in different years”;
}
```

In main(), the statement if(my_age==your_-age) has the expression my_age==your_age as conditional expression. If my_age and your _age is equal to the return is true(i.e) value is one(1), otherwise the return is not true(ie) the value is zero(0).

LOGICAL OPERATORS:

Any expression that evaluates to zero denotes a false logical condition, and that evaluating to non zero value denotes a true logical condition. Logical operators are useful in combining one or more conditions.

OPERATOR	MEANING
&&	Logical AND
||	Logical OR
!	Logical NOT

The first two operators && and || are binary where as the exclamation is a binary operator and is used to negate a condition.

The result of logical operations when applied to operands with all possible values.

LOGICAL AND:

(e.g) a>b&&==0

The expression on the left is a>b and that on the right is x==10. The whole expression values to true only if both expressions are true.

OPERAND-1 a	OPERAND-2 b	a&&b	a || b	~a	~b
F	F	F	F	T	T
F	T	F	T	T	F
T	F	F	T	F	T
T	T	T	T	F	F

LOGICAL OR(||):

a <m || a<n

The expression is true if one of them is true or both of them are true.

LOGICAL NOT(!)

The NOT operator takes a single expression and evaluates to true if the expression is false and evaluates o false if the expression is true.

The unary negation operator has a higher precedence amongst these, followed by the logical AND(&&) operator and then the logical OR(||) operator and are evaluated from left to right.

BITWISE OPERATORS:

OPERATOR	MEANING
&	Bitwise AND
|	Bitwise OR
^	Bitwise EX-OR
~	Bitwise complement
<<	Shift left
>>	Shift right

The support of Bitwise manipulation on integer operands is useful in various applications.

Bitwise AND

C=a&b;

int a=13

Int b=7

The variables a, b and c as 16 bit integer.

a= 0000 0000 0000 1111

b= 0000 0000 0000 0111

a&b= 0000 0000 0000 0101 c value is = 5.

BITWISE OR

c= a|b

a= 0000 0000 0000 1101

b= 0000 0000 0000 0111

a|b= 0000 0000 0000 1111 c value is = 15.

BITWISE XOR:

C=a∧b;

After this statement is executed, a bit in c will be 1 whenever the corresponding bits in a and b differ.

a= 0000 0000 0000 1101

b= 0000 0000 0000 0111

a∧b= 0000 0000 0000 1010 c value is = 10.

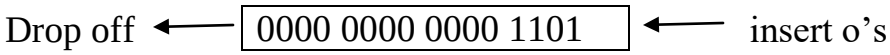
SHIFT OPERATORS:

LEFT SHIFT OPERATOR

C= a<<3

The value of integer is shifted left by three bit positions. The result is assigned to the integer c. since the value of a is 0000 0000 0110 1000(104 in decimal)

Left shift<<



After left bit shift by 3 places.

A<<3

0000 0000 0110 1000

The three left most nits drop off due to the left shift. The zero's are inserted in the right.

RIGHT SHIFT OPERATOR:

C=a>>2

The value of a is shifted right by 2 positions

The value of a is 0000 0000 0000 1101

The value of c after the execution of the above statement is 0000 0000 0000 0011

The value of c is (in decimal)= 3

COPOUND ASSIGMENT OPERATORS

The assignment operator =(equal sign) evaluates the expression on the right and assigns the resulting value to the variable on the left.

Variable operator = expressions/ constant/ function;

i=i+10;

if can be rewritten in the compact from the follows:

i+=10;

the operator += is known as compound assignment operator.

INCREMENT AND DECREMENT OPERATORS:

++ is known as increment and – is known as decrement operator.

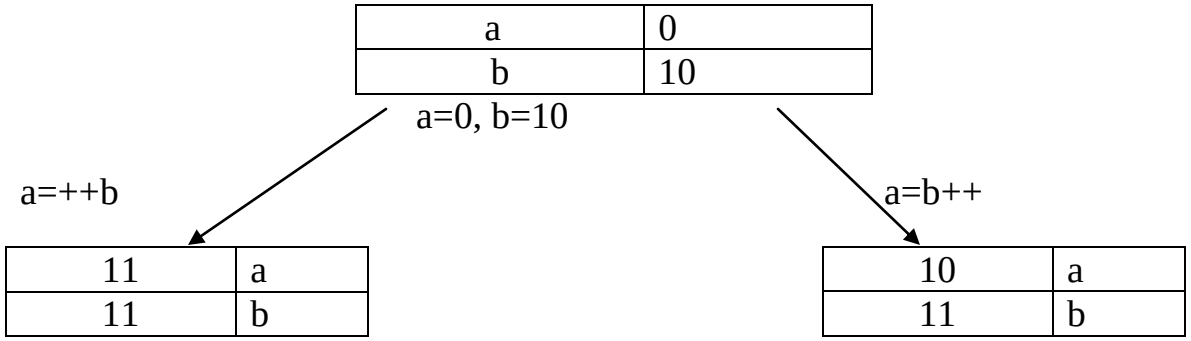
These operators increase or decrease the value of a variable on which they operate by one.

SYNTAX

++ variable -- variable

Variable ++ variable—

The prefix and postfix for increment expressions ++m and m++



CONDITIONAL OPERATORS (TERNARY OPERATORS)

An alternative method to using a simple if else construct is the conditional expression operator. It is called the ternary operators, which operates on three operands.

Expression 1 ? expression2 : expression 3

Z= (a>b) ? a: b

Z= ((num%2)? “ODD”: “EVEN”)

CONTROL FLOW

BRANCHING STATEMENTS

(a) If statement	Conditional statement
(b) If else statement	Conditional statement
(c)switch statement	Conditional statement
(c) Goto statement	Unconditional statement

LOOPING STATEMENTS

Loop cause a section of code to be executed repeatedly until a termination condition is met.

(a) For statement
(b) While statement
(c) Do-while statement

BRANCHING STATEMENTS

IF STATEMENTS

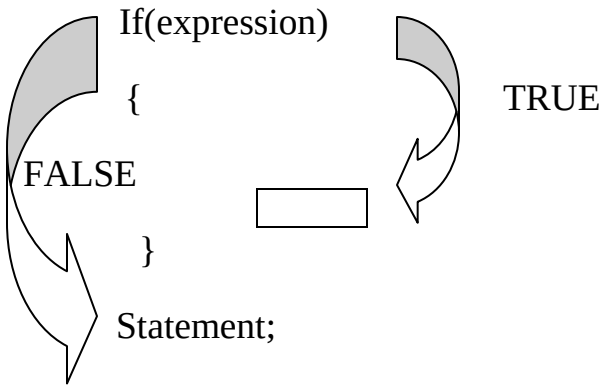
The if construct is a powerful decision making statement which is used to control the sequence of the execution of statement.

SYNDAX:

If (test expression)

Statement;

The test expression should always enclosed in parentheses. If test_expression is true then the statement immediately following it is executed otherwise, control passes to the next statement following the if construct.



IF_ELSE STATEMENT: