UNIT-2

Module – 2 : (Arithmetic expressions & Conditional Branching)

- Arithmetic expressions and precedence
- Operators and expression using numeric and relational operators
- Mixed operands, Type conversion
- Logical operators, Bit operations
- Assignment operator, Operator precedence and associativity.

Conditional Branching:

- Applying if and switch statements
- Nesting if and else
- Use of break and default with switch.

Short Question & Answers

Ques 1 What are C expressions? Give examples.

Ans : An expression is a combination of variables, constants, operators and function call. It can be arithmetic, logical and relational for example:-

```
int z = x+y
// arithmatic expression


a>b          // relational
a= =b        // logical
func(a, b)   // function call
```

Expressions consisting entirely of constant values are called *constant expressions*. But if i were declared to be an integer variable, the expression 180 + 2 – j would not represent a constant expression.

Ques : 2 Define the term C operator. Explain Arithmetic and Logical Operators.

Ans: An operator is a symbol which helps the user to command the computer to do a certain mathematical or logical manipulations. Operators are used in C language program to operate on data and variables.

(i) Arithmetic Operators

All the basic arithmetic operations can be carried out in C. Example : +, -, %, / , *, &&, etc.Both unary and binary operations are available in C language. Unary operators operate on a single operand, where operand can be a constant, a variable or a valid C expression. Binary operators operate on two operands. Ternary operators operate on three operands.

***Unary Operators*:** +a (unary plus), -a (unary minus), ++a (pre increment) , !a(logical NOT).

***Binary Operators*:** a+b (binary addition), a-b(binary subtraction),a*b(multiplication),

a / b (division), a % b (Modulus division), x > = y (relational operator)

Ternary Operator: This is also called conditional operator (? :)

Test expression ? expression 1 : expression 2

Logical Operators: C has the following logical operators; they compare or evaluate logical and relational expressions. The output evaluated is either a true value or false value of an expression. Represented as 0 (false) , 1(True).

A || b is Logical OR, a &&b is Logical AND , !a is Logical NOT

Ques 3. What is the difference between "=" and "= ="?

Ans: **= (assignment operator)** in C language is used to assign the value of right-hand side value/variable/expression to the left hand side variable .e.g: a = 4; here constant 4 at R.H.S is called r-value and identifier a at the L.H.S is called l-value

= = (Equality Operator)in C language is used to check the value of left hand variable/expression with the right hand Variable/expression. Whether the two values are equal or not. It returns true if these are equal else it will return false.

e.g. (2==3) : It means whether 2 is equal to 3 or not which is false so output is 0.

(3==3) : It mean 3 is equal to 3 ,which is true so output is 1

Ques 4. What is the output of arithmetic statement, S = 1/3 * a /4 – 6/ 2 + 2 / 3 * 6 / g ?Given that (int a = 4 , int g = 3 , assume s to be an int type).HINT:** apply the concept of type casting.

Ans : S= 1/3 * a/4 – 6/2 + 2 /3 * 6/g

= 1 /3 * 4 /4 -6 /2 + 2/3 * 6 /3

= 0 * 4 / 4 – 6 /2 + 2 / 3* 6 / 3

= 0 / 4 – 6 /2 + 2 /3 * 6 /3

= 0 – 3 + 0 * 2

= -3

Ques 5 . Give the outpu of following code :

```
void main()
   {   float u = 3.5  ;
       int v , w , x , y ;
        v = ( int ) ( u + 0.5) ;
        w = (int)  u + 0.5 ;
        x = (int) ( ( int) u + 0.5) ;
        y = ( u + (int) 0.5 ) ;
      printf ( " %d %d %d %d " , v , w , x , y )  ;
   }
```

Ans : Output is : 4 3 3 3

Ques 6. What is the difference between the pre-increment operator and the post-increment Operator? Give example also.

Ans. (i) The pre-increment operator increases the operand's value by 1 first, and then returns the modified Value.

(ii) The post-increment operator stores a copy of the operand value in a temporary Location and then increases the operand value by 1. For example given x = 1, the ++x expression returns 2, while the x++ expression returns1.

EXAMPLE: What is the output of the following C code?

```
main()
{   int i =5 , ;
    printf ( " %d %d %d %d %d " , i++ , i-- , ++i , --i , i ) ;
}
```

Ans: **4 5 5 4 5**

Ques 7. Find the output of the following program?

```
main( )
{
int x=100;
printf("%d",10 + x++);
printf("%d",10 + ++x);
}
```

Ans. 110 112

Ques 8. What is the output of the following C code?

```
main()
{   int i = 0 , j = 1 , k =2 , l ;
    l = i && j++ && ++k ;
    printf ( " %d %d %d %d " , i , j , k , l ) ;
}
```

Ans: Output : 0 1 2 0

Ques 9. What will be the value of x when the following segment is executed?

```
int x=10, y=15;
x=(x<y) ? (y+x) : (y-x) ;
```

Ans. Output – 25

Ques 10. What is the dangling else problem?

Ans: When a program contains more number of if clauses than else clauses, then their exists a potential ambiguity regarding with which if clause does not match with else clause. This ambiguity is called ***Dangling else problem***

Ques 11. What is the precedence and associativity of operators?

Ans : **Precedence :** When an expression hasb more than one operators then relative priorities of operators with respect to each other is checked to evaluate expression in that order.

E.g : Multiplicative operators (* , /, %) have high priority than additive operators (+ -).

Note : If all the operators in same expression have equal priority then operator which comes first from left is evaluated first and then we move towards right .

Exmaple : In expression 2 / 3 * 6 % 4 , all are multiplicative operators, hence have same precedence. So first / is evaluated , then * is evaluated and then % division is performed.

Associativity: This defines the direction in which the operator acts on the operands. It can be either left to right or right to left.. First precedence is checked then associativity is applied.

++ (prefix /postfix)	**Right to left**
***, / , %**	**Left to right**
< , < =, > , >=	**Left to right**
= assignment	**Right to left**
&& , ||	**Left to right.**

Long Questions & Answers

Ques 12. Describe the bitwise operators available in C with examples.

Ans: These operators work with bits rather than larger structures such as a byte. For example, each of the eight bits in a byte can be used as an individual flag to signal yes/no, on/off (1 or 0) about some condition. The Boolean operators AND, OR and NOT also deal with individual bits rather than bytes. Bitwise operators are programming commands that work with individual bits. The primary ones are.

Symbol	Function
<<	Shift left
>>	Shift right
&	bitwise AND
|	bitwise OR
^	bitwise XOR (Exclusive OR)
~	bitwise NOT (0 to 1; 1 to 0)

Examples : (i) int a = 10 , b = 20 , c = 0;

C = a & b ;

Here compiler will first convert int variables a and b in binary form , then bit wise AND is performed.

(ii) **Shift operator** shift the bits either to left or right. The syntax of shift operation can be given as:

Operand op num , Where the bits in operand are shifted left or right depending on the operator (left if operator is << and right if operator is >>) = 00011101

X << 1 produces 00111010

Shifting once to the left multiplies the number by 2. Hence multiple shifts of 1 to the left result in multiplying by 2 several times. If x = 0001 1101 then x >>4 produces 0000 0001. Shifting from right by 1 bit divides the number by 2.

Ques 13. What is meant by type conversion and type casting? Why it is necessary? Explain about implicit and explicit type conversion with examples.

Ans : When any C expression or statement involves different data types e.g : multiplication of a float number by integer number , then to get the correct result in terms of values ***Type conversion / Type casting is required in programming.*** In this we change a variable of one data type to variable of another data type.

** Type conversion is done implicitly by the compiler

** Type casting is performed explicitly by the programmer.

(a) **Type conversion** : It is done when expression has variables of different data types. To evaluate the expression , data type is promoted from lower to higher data type (or from higher to lower). Type conversion is automatically performed.

e.g : float x ;

int y = 3;

x = y ; // Here x will have the value 3.0 as automatically integer value is converted to its equivalent floating point value.

Conversion hierarchy of data types is as shown below :

long double	Higher level
double	
Float	
Unsigned long int	
Long int	
Unsigned int	
Int	
Short , char	Lower level

** When a char type is operated with an int type char is promoted to int.

** When a float type data is operated with an int , then int is promoted to float.

** When any operand is double then another operand is also converted to double

(b) **Type casting :** It is also called as forced conversion. It is done when value of higher data type is to be converted to lower data type. This conversion is under programmer's control , not in compiler's control.

Demotion : float f = 3.5 ;

int I ;

I = f ;

Statement I = f results in f to be demoted to type int , i.e. the fractional part of f will be lost and I will contain 3 not 3.5 ; this is called down conversion/demotion. But result is not correct so explicit type

casting is required. **Explicit type casting is done by placing the target data type in parantheses followed by the variable name that has to be converted.**

Variable 2 = (data type) variable1

e.g : float salary = 10000.00 ;

int tot_sal =0;

sal = (int) salary

** When floating point numbers are converted to integers , the digits after decimal are truncated.

Ques 14. What is conditional branching statement ? Explain switch case statements in detail , with a proper example.

Ans : The conditional branching statements help to jump from one part of the program to another depending whether a particular condition is true or false. These decision control statements include :

(i) if statement (ii) if – else statement (iii) if – else-if statement (iv) switch statement.

A switch case statement is a multi-way decision statement that is simplified version of if – else block. Staement blocks refer to statements lists that may contain zero or more statements. These statements ar not enclosed within opening and closing braces.

Switch statements are mostly used in following situations :

(a) When there is only one variable to evaluate in the expression.

(b) When many conditions are being tested for.

(c) Switch case is preferable over if –else if many conditions are to be tested.

Syntax of switch – case statement

```
Switch ( int expression/ int variable)
    {
     Case  value1 :
          Statement  Block 1 ;
          Break ;
       Case value 2 :
            Statement block 2;
           Break;
…….
  Case value N :
```

```
        Statement block N ;
        Break ;
    Default :
        Statement block D ;
        Break ;
    }
Statement x ;
```

Switch case compares the value of integer expression/variable with the value of each case statement that comes in that case.If value of switch expression and case number matches then that statement block is executed.

Default Case : When switch expression does not matches with any case value , then statements in default case are executed. (default case is optional).

Use of break in switch case : The break statement must be used after each case statement in ideal conditions. If it is not used then the case which is matched and all the cases following that case are executed resulting in logical error or incorrect output.

Some Rules of Switch – Case statements

(i) Control expression inside switch must be integer type variable or integeral expression
E.g : switch (int n) , switch (int x + 6) etc.

(ii) Each case label should be followed by a constant or a constant expression.

(iii) Every case label must have unique constant expression value.

(iv) Case labels must end with a colon.

(v) Default label can be placed anywhere in switch block. There must be only one default in switch program.

(vi) case labels should not be logical or relational expressions.

(vii) Order in which case labels are written are not fixed.

Example 1 of switch case with break	Example 2 of switch case without break
# include <stdio.h> # include <conio.h> Void main() { char grade = 'C ' switch (grade) { Case 'O ' : Printf (" \n Outstanding ") ;	# include <stdio.h> Int main() { Int option =1; Switch (option) { Case 1 : printf (" I am in case 1\n") ; Case 2 : printf (" I am in case 2\n") ;

```
  Break ;
Case ‘ A’ :
   Printf ( “ \n Excellent ”) ;
  Break ;
Case ‘B’ :
   Printf ( “ \n GOOD ”) ;
  Break ;
Case ‘C ’ :
   Printf ( “ \n Satisfactory ”) ;
  Break ;
Case ‘F ’ :
   Printf ( “ \n Fail ”) ;
  Break ;
Default :
   Printf ( “ \n Outstanding ”) ;
  Break ;
}  getch (); } // End of main()
**************************************
```

Output : Satisfactory

```
   Default : printf (“ I am in case default ”) ;
}
  return 0;
  getch();
}
```

Output : I am in case 1
I am in case 2
I am in case default.

Ques 15. What is the meaning of statement in C language. Briefly explain its classification with example.

Ans : A statement is the smallest logical unit that can independently exist in a C program. No entity smaller than statement (i.e variables, expressions, constants etc) can exist independently in a program. Example : a= b + 5 is an expression , but when it is terminated by a semicolon , like a = b + 2 ; then it becomes a assignment statement. It can be also called arithmetic statement , because operator + is also associated here.

Classification of C statements : (a) Based upon the type of action they perform.
(b) Based upon the number of constituent statements.
(c) Based upon their role.

Based upon the type of action they perform

1. **Non executable statements** **2. Executable statements**

   Non executable statements tell compiler how to build a program, how to allocate memory, interpret and compile other statements. A non-executable statements can affect only statements below them , because compiler starts scanning from top to bottom.

Examples : Prototype declarations, global variables, preprocessor directives , header files.

Executable statements: These represent the instructions that are to be performed when the code is executed. Executable statement appears only inside the function body. ***Example : looping statement, branching statements, assignment statements .***

Based upon the number of constituent statements

1. Simple statement 2, Compound statement

Simple statements are those which have only single statement, terminated by a semicolon.

Example : int a = 10; // declaration statement

a = a + 10 // assignment statement.

printf(" Welcome to the concepts of C ") ; // print statement.

A compound statement contains sequence of more than single statements enclosed within a pair of curly braces {}.Example :

```
{
    int x =10;
    x = x + 3 ;
    ++x ;
    printf( " %d" , x ) ;
}
```

A compound statement is also called as block. Compound statement can be empty aslo, i.e nothing written inside the braces. E.g : { }. A compound statement need not to be terminated by a semicolon after braces.

Based upon their role

1. Declaration statement and Definition statement.
   Example: int a ; is a declaration. But if we assign some value to this variable it becomes definition of this variable. E.g : int a = 30 ;
2. Null statement and expression: Null statements are those which contain only semicolon, nothing else. Expression is 2 + 3 / 5.
3. Flow control statements: The order in which control flow occurs in the program is called flow of program control. Program control flows sequentially from top to bottom. These are of two type :
   (A)Branching Statement: (a) Selection Statements (b) Jump statements.
   (B) Iteration Statements

Examples : Selection statements : if statement, if-alse statement, switch statement.
Jump Statements : break ,continue, return , goto.
Iteration Statements : for loop, while loop, do-while loop.

Important Programs

1. **Program to find the simple interest, given principle, rate of interest and time.**	2. **Ramesh's basic salary is input through the keyboard. His dearness allowance is 40% of basic salary, and house rent allowance is 20% of basic salary. Write a program to calculate his Gross Salary.**
#include <stdio.h> #include <conio.h> void main() { float p, r, si; int t; clrscr(); printf("Enter the values of p,r and t\n"); scanf ("%f %f %d", &p, &r, &t); si = (p * r * t)/ 100.0; printf ("Amount = Rs. %5.2f\n", p); printf ("Rate = Rs. %5.2f%\n", r); printf ("Time = %d years\n", t); printf ("Simple interest = %5.2f\n", si); getch() ; }	#include<stdio. #include<conio.h> void main() { float bs , da= 0.0, hra=0.0 ,gross=0.0 ; clrscr() ; printf("Enter the basic salary\n") ; scanf("%f" ,&bs) ; printf("BASIC SALARY OF RAMESH=%f\n ", bs) ; da = (bs*40) / 100 ; hra = (bs *20) /100 ; gross = bs + da + hra ; printf("**********"); printf("GROSS SALRY OF RAMESH = % f ", gross) ; getch() ; }

3. Program to find the area of a circle, given the radius.

```
#include <stdio.h>
#include <conio.h>
#include <math.h>

#define PI 3.142
void main()
{
   float radius, area;
   clrscr();
  printf("Enter the radius of a circle\n");
  scanf ("%f", &radius);
  area = PI * pow (radius,2);
 printf ("Area of a circle = %5.2f\n", area);
 getch() ;
 }
```

4. Program to find the area of a triangle, given three sides a, b and c.

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
 void main()
 {  float  a, b, c ;
     float s =0.0 , area =0.0 ;
     clrscr();
printf("Enter the values of a,b and c\n");
scanf ("%f %f%f ", &a, &b, &c);
   s = (a + b + c) / 2;                    /*
computes perimeter */
  area = sqrt ( s * (s-a) * (s-b) * (s-c));          /*
compute the area *
 printf ("Area of a triangle = %f\n", area);
```

5. Program to swap the values of two variables without using third variable.

```
#include<stdio.h>
#include<conio.h>
 void main()
  {
     int  a , b ;
    clrscr() ;
  printf("Enter the values of a and b \n") ;
  scanf("%d %d" ,&a , &b) ;
  printf("Values variables before swapping
          are a=%d \t b=%d: \n", a , b) ;
        a =  a + b ;
        b = a – b ;
        a =  a – b ;
printf("Values of a and b after swapping
        are:\n") ;
printf("a=%d\t\t b=%d",a ,b) ;
getch();  }
```

6. Program to multiply a given number with 2^n ,without using multiplication operator.

```
#include<stdio.h>
#include<conio.h>
void main()
 {
    int num , n , result ;
    clrscr() ;
    printf("Enter the number to be
            multiplied \n") ;
    scanf("%d" ,&num) ;
    printf("Enter the value of n \t") ;
    scanf("%d", &n) ;
    result = num << n ;
    printf("Result of multiplication = %d",
            result);
    getch() ;
}
```

7. **Program for finding the sum of a five digit no.**	8. **. Write a C program to check whether a given integer is odd or even.**
#include<stdio.h> #include<conio.h> void main() { long int num , sum =0 , rem ; **// Max. five digit no. can be 99999, so long int declared** clrscr() ; printf(“Enter the five digit number\n”) ; scanf(“% ld” , &num) ; printf(“The five digit number=%ld\n”, num) ; rem1 = num%10 ; num = num /10 ; rem2 = num%10 ; num = num /10 ; rem3 = num%10 ; num=num/10 ; rem4 = num%10; num=num/10 ; sum = rem1 + rem2 + rem3 + rem4 + num ; printf(“Sum of the digits of %ld =% ld”, num ,sum) ; getch(); }	#include <stdio.h> #include <conio.h> void main() { int x , remainder; clrscr(); printf("Enter an integer :"); scanf ("%d", x); remainder = x % 2; if (remainder == 0) { printf ("%d, is an even integer\n", x); else printf ("%d, is an odd integer\n", x); } getch (); } /* End of main() */

9. Write a C program to find the largest of three numbers.	**10. Program to determine whether a triangle is right angle , scalene, equilateral or isosceles where the Sides a , b and c are entered by the user. Also check whether the triangle is possible or not?**
#include <stdio.h> #include <conio.h> #include <math.h> void main() { int a, b, c; clrscr(); printf("Enter the values of a,b and c\n"); scanf ("%d %d %d", &a, &b, &c); printf ("a = %d\tb = %d\tc = %d\n", a,b,c); if (a > b) { if (a > c) { printf ("A is the greatest among three\n"); } else { printf ("C is the greatest among three\n"); } } else if (b > c) { printf ("B is the greatest among three\n"); } else printf ("C is the greatest among three\n"); }	#include<stdio.h> #include<conio.h> void main() { float a , b , c ; clrscr() ; printf ("Enter the sides of the triangle : \n") ; scanf("%f %f %f " , & a , & b , &c) ; if ((a + b < c) || (b + c < a) || (a + c < b)) **printf ("Triangle is NOT POSSIBLE") ;** if ((a * a = = b * b + c * c) || (b * b = = a * a + c * c)) || (c * c = = a * a + c * c)) ; **printf(" Triangle is RIGHT ANGLED") ;** if ((a = = b) && (b != c)) ; **printf("Triangle is ISOSCELES") ;** if ((b = = c) && (c!= a)) ; **printf("Triangle is ISOSCELES") ;** if ((c = = a) && (a != b)) ; **printf("Triangle is ISOSCELES") ;** if ((a = = b) && (b = = c)) ; **printf(" Triangle is EQUILATERAL") ;** if ((a != b) && (b != c) && (c!=a)) ; **printf(" Triangle is SACLENE") ;** getch () ; }

[END OF 2nd UNIT]