UNIT-II

Unit-2 Question bank

1. (a) What is Shift operation? Perform different types of shift operations on 101110011.
ANS:

Right Shift Operator

Right shift operator shifts all bits towards right by certain number of specified bits.

It is denoted by >>.

```
212 = 11010100 (In binary)

212>>2 = 00110101 (In binary) [Right shift by two bits]

212>>7 = 00000001 (In binary)

212>>8 = 00000000

212>>0 = 11010100 (No Shift)
```

Left Shift Operator

Left shift operator shifts all bits towards left by a certain number of specified bits.

The bit positions that have been vacated by the left shift operator are filled with 0.

The symbol of the left shift operator is <<.

```
212 = 11010100 (In binary)

212<<1 = 110101000 (In binary) [Left shift by one bit]

212<<0 = 11010100 (Shift by 0)

212<<4 = 110101000000 (In binary) =3392(In decimal)
```

Example : Shift Operators

```
#include <stdio.h>
void main()
{
  int num=212, i;
  for (i=0; i<=2; ++i)
    printf("Right shift by %d: %d\n", i, num>>i);

  printf("\n");

  for (i=0; i<=2; ++i)
    printf("Left shift by %d: %d\n", i, num<<i);

}
```

1. (b) Write a C program to print. And also explain the looping concepts used in it.

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

ok

```
#include<stdio.h>
void main()
{
int i,j;
for(i=1;i<=5;i++)
{
  for(j=1;j<=i;j++)
   {
      printf("%d  ",j);
   }
printf("\n");
}
```

2. Why switch case is better than nested if? Why you need break, default statement in switch case? Explain with an example program.

Sol:

A switch statement is usually more efficient than a set of nested ifs

Switch is better than if else if

1. Check the Testing Expression: An if-then-else statement can test expressions based on ranges of values or conditions, whereas a switch statement tests expressions based only on a single integer, enumerated value, or String object.
2. Clarity in readability: A switch looks much cleaner when you have to combine cases. Ifs are quite vulnerable to errors too. Missing an else statement can land you up in havoc.
3. Adding/removing labels is also easier with a switch and makes your code significantly easier to change and maintain.

break Statement

In C programming, break is used in terminating the loop immediately after it is encountered. The break statement is used with conditional if statement.

Syntax :

```
break;
```

The break statement can be used in terminating all three loops for, while and do...while loops. The break; statement is also used in switch statement to exit switch statement.

Default case: if no case constant is matched with switch expression then Default case will execute

/*C program to design calculator with basic operations using switch.*/

```
#include <stdio.h>
void main()
{
int num1,num2;
float result;
char ch;
printf("Enter first number: ");
scanf("%d",&num1);
printf("Enter second number: ");
scanf("%d",&num2);
printf("Choose operation to perform (+,-,*,/,%): ");
scanf(" %c",&ch);
switch(ch)
```

```
{
case '+':
        result=num1+num2;
        break;
case '-':
        result=num1-num2;
        break;
case '*':
        result=num1*num2;
        break;
case '/':
        result=(float)num1/(float)num2;
        break;
case '%':
            result=num1%num2;
            break;
default:
            printf("Invalid operation.\n");
}
printf("Result: %d %c %d = %f\n",num1,ch,num2,result);

}
```

3.a) In detail, discuss various sized integer types supported by C. Give their storage and value ranges

Data Type	Memory (bytes)	Range	Format Specifier
short int	2	-32,768 to 32,767	%hd
unsigned short int	2	0 to 65,535	%hu
unsigned int	4	0 to 4,294,967,295	%u
int	4	-2,147,483,648 to 2,147,483,647	%d
long int	4	-2,147,483,648 to 2,147,483,647	%ld
unsigned long int	4	0 to 4,294,967,295	%lu
long long int	8	-(2^63) to (2^63)-1	%lld
unsigned long long int	8	0 to 18,446,744,073,709,551,615	%llu

3(b) Write a C program to determine the character entered by user. And also explain the control structures used in it. .

Sol:

```
#include <stdio.h>
void main()
{
   char ch;

   //Asking user to enter the character
   printf("Enter any character: ");
   //storing the entered character into the variable ch
   scanf("%c",&ch);

   if( (ch>='a' && ch<='z') || (ch>='A' && ch<='Z'))
     {
        printf("The entered character %c is an Alphabet",ch);
     }
   else if(ch>='0'||ch<='9')
     {
        printf("The entered character %c is a number",ch);
     }
   else
     {
         printf("The entered character %c is not an alphabet or number",ch);
     }
}
```

4. (a) Explain pre and post tests used in repetition. Write a C program to print the average of numbers entered by the user.

Sol:

Pre-test Loop A pre-test loop is one in which the loop condition is tested before entering the loop.

example: while loop and for loop

Post-test Loop: In a post-test loop, the test of the loop condition occurs after the body of the loop has been carried out one time. So, the statements in the body of the loop are always executed at least one time.

ex: do while loop

C program to print the average of numbers entered by the user.

```
#include<stdio.h>
void main()
{
   int n,i,sum=0,avg;
   for(i=1;i<=10;i++)
     {
        printf ("enter the value of n");
        scanf(""%d",&n);
        sum=sum+n;
     }
```

```
    avg=sum/10;
    printf("avg=%d",avg);
}
```

4 (b) Write a program to classify a given number is prime or composite using for loop

Sol:.

```
#include<stdio.h>
void main()
{
int num,count=0,i;
printf("enter a number");
scanf("%d",&num);
 for(i=1;i<=num;i++)
   {
      if(num%i==0)
        {
          count=count+1;
        }
   }
 if(count==2)
 {
   printf(" %d is prime",num);
 }
 else
 {
   printf(" %d is  composite",num);
 }
 }
```

ok

5.(a) Give syntax for conditional branching statement? Write an example program to demonstrate nested if statement?

Sol:

ok

if statement

This is the most simple form of the branching statements.

It takes an expression in parenthesis and an statement or block of statements. if the expression is true then the statement or block of statements gets executed otherwise these statements are skipped.

NOTE: Expression will be assumed to be true if its evaulated values is non-zero.

if statements take the following form:

```
if (expression)
  {
    Block of statements;
  }

or

if (expression)
  {
    Block of statements;
  }
else
```

```
  {
    Block of statements;
  }

or

if (expression)
  {
    Block of statements;
  }
else if(expression)
  {
    Block of statements;
  }
else
  {
    Block of statements;
  }
```

c program to implement nested if

```
#include<stdio.h>
void main()
{
   int p,q,r,s;
   printf ("enter p,q,r,s values");
  scanf('%d%d%d%d",&p,&q,&r,&s);
  if(r>0&&s>0)
   {
       if(p%2==0)
        {
         if(q>r&&s>p)
             {
               if(r+s>p+q)
                {
                    printf("correct values");
                }
             }
        }
    }
else
  {

    printf("wrong values");
  }
}
```

5(b) What is the use of break and continue statements? Explain with an example

Sol:

BREAK & CONTINUE STATEMENT:

There are two statements built in C programming, break; and continue; to alter the normal flow of a program.

break Statement

In C programming, break is used in terminating the loop immediately after it is encountered. The break statement is used with conditional if statement.

Syntax :

```
break;
```

The break statement can be used in terminating all three loops for, while and do...while loops.

The break; statement is also used in switch statement to exit switch statement.

CONTINUE STATEMENT

It is sometimes desirable to skip some statements inside the loop. In such cases, continue statements are used.

Syntax of continue Statement

```
continue;
```

Just like break, continue is also used with conditional if statement.

```
/*C program to design calculator with basic operations using switch.*/
#include <stdio.h>
void main()
{
int num1,num2;
float result;
char ch;
 printf("Enter first number: ");
scanf("%d",&num1);
printf("Enter second number: ");
scanf("%d",&num2);
printf("Choose operation to perform (+,-,*,/,%): ");
scanf(" %c",&ch);
switch(ch)
{
case '+':
        result=num1+num2;
        break;
case '-':
        result=num1-num2;
        break;
case '*':
        result=num1*num2;
        break;
case '/':
        result=(float)num1/(float)num2;
        break;
case '%':
           result=num1%num2;
           break;
default:
           printf("Invalid operation.\n");
}
printf("Result: %d %c %d = %f\n",num1,ch,num2,result);

}
```

Notes better

OK

6 (a) Explain the syntax of event and counter controlled loops used in C language with an example.

Sol:

Event controlled loop

OK

1.here we don't know how many times loop is going to work

2. An event changes loop expression to be evaluated from true to false.

3.to implement Event controlled loops we will use while or do while loops

Example: when we read the data loop expression may be evaluated to true. but, we encounter the end of the input. loop expression may be evaluated to false .

example2: palindrome, finding lcm and gcd are some of the examples for event control loop programs

Write a c program to find given no is Armstrong or not.

```
#include<stdio.h>
void main()
{
    int num,sum=0,num1,rem;
   printf("enter the number");
   scanf("%d",num);
  num1=num;
  while(num>0)
       {
         rem=num%10;
         sum=sum+rem*rem*rem;
         num=num/10;
       }
if(num1==sum)
       {
          printf("%d is armstrong number",num1);
       }
}
```

Counter controlled loop:

1.here we know how many times loop is going to work.

2.we will for loop to implement counter controlled loop

example: sum of n natural numbers, Fibonacci series these are examples for counter controlled loop programs.

Write a c program to find the factorial of a given number

```
#include<stdio.h>
void main()
{
    int num,fact=1,i;
   printf("enter the number");
   scanf("%d",num);
for(i=1;i<=num;i++)
    {
         fact=fact*i;
    }
       printf("factorial=%d",fact);
}
```

6.b) Write a C program to read numbers until -1 is encountered and also count the positive, negative and zeros encountered by the users using while loop construct.

```
#include<stdio.h>
void main()
{
```

```
int num,pc=0,nc=0,zc=0;
do
{
printf("enter numer");
scanf("%d",&num);
if(num>0)
        pc=pc+1;
if(num<0)
        nc=nc+1;
if(num==0)
        zc=zc+1;
}while(num!=-1);
printf("pc=%d    nc=%d      zc=%d",pc,nc,zc);
}
```

7.Explain concept of loop? what are the different types of loops used in c along with syntax and example?
ANS:

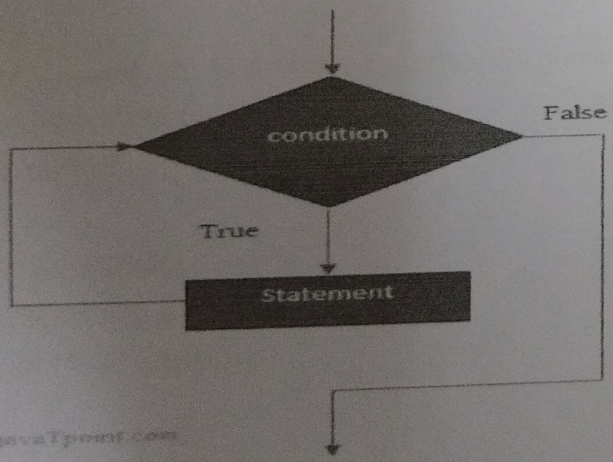
while loop in C

While loop is also known as a pre-tested loop. In general, a while loop allows a part of the code to be executed multiple times depending upon a given boolean condition. It can be viewed as a repeating if statement. The while loop is mostly used in the case where the number of iterations is not known in advance.

Syntax of while loop in C language

```
1. while(condition)
2. {
3. //code to be executed
4. }
```

Flowchart of while loop in C



write a c program to find the factorial of a given number using while loop

```
#include<stdio.h>
void main()
{
 int num,i,fact;
printf("enter a number");
```

```
scanf("%d",&num);
i=1;
while(i<=n)
{
  fact=fact*i;
   i++;
}
    printf('factorial=%d",fact);
}
```

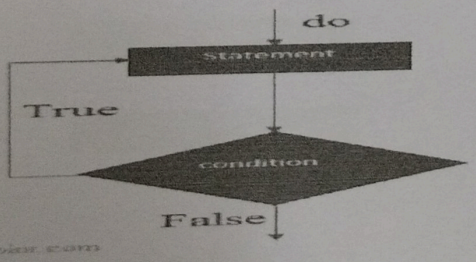
do while loop in C

The do while loop is a post tested loop. Using the do-while loop, we can repeat the execution of several parts of the statements. The do-while loop is mainly used in the case where we need to execute the loop at least once. The do-while loop is mostly used in menu-driven programs where the termination condition depends upon the end user.

do while loop syntax

The syntax of the C language do-while loop is given below:

1. **do**{
2. //code to be executed
3. }**while**(condition);



write a c program to find the factorial of a given number using do while loop

```
#include<stdio.h>
void main()
{
 int num,i,fact;
printf("enter a number");
scanf("%d",&num);
i=1;
do
{
  fact=fact*i;
   i++;
} while(i<=n);

    printf('factorial=%d",fact);
}
```

for loop in C

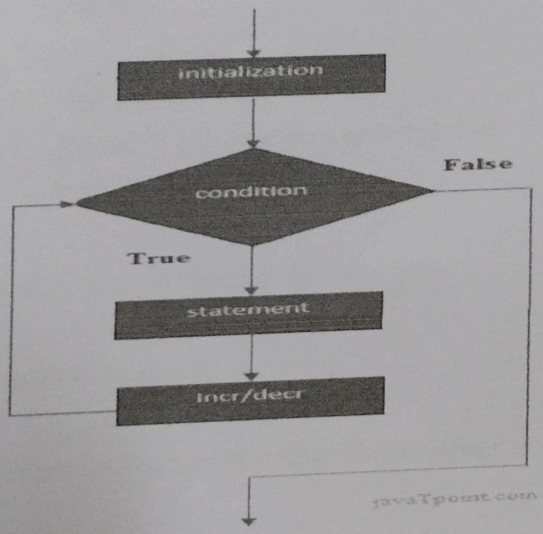
The **for loop in C language** is used to iterate the statements or a part of the program several times. It is frequently used to traverse the data structures like the array and linked list.

Syntax of for loop in C

The syntax of for loop in c language is given below:

```
1. for(Expression 1; Expression 2; Expression 3)
2. {
3. //code to be executed
4. }
```

Flowchart of for loop in C



write a c program to find the factorial of a given number using for loop

```
#include<stdio.h>
void main()
{
 int num,i,fact;
printf("enter a number");
scanf("%d",&num);
for(i=1;i<=n;i++)
{
 fact=fact*i;
}
   printf('factorial=%d",fact);
}
```

8.a) Explain various bitwise operators in C

Sol:

Operator	Description	Example	Same as	Result	Decimal
&	AND	x = 5 & 1	0101 & 0001	0001	1
|	OR	x = 5 | 1	0101 | 0001	0101	5
~	NOT	x = ~ 5	~0101	1010	10
^	XOR	x = 5 ^ 1	0101 ^ 0001	0100	4
<<	Left shift	x = 5 << 1	0101 << 1	1010	10
>>	Right shift	x = 5 >> 1	0101 >> 1	0010	2

8 (b) Write a C program to find the sum of N natural numbers using for loop

```
#include <stdio.h>
void main()
  {
        int num, i, sum = 0; //
       printf(" Enter a positive number: ");
       scanf("%d", &num);

  for (i = 0; i <= num; i++)
  {
    sum = sum + i;
  }
  printf("\n Sum of the first %d number is: %d", num, sum);
}
```

OR

9Q) What is a flow control statement? Explain various Two-way Selective statements in C with example each

ANS: Control flow statements enable us to specify the flow of program control. i.e: the order in which the instructions in a program must be executed. They make it possible to make decisions, to perform tasks repeatedly or to jump from one section of code to another

there are 3 types control statements in c:

1.Decision making and selection statements

2.Iteration or repetitive statements

3.JUMP statement

2-way selection statements are

1.if else

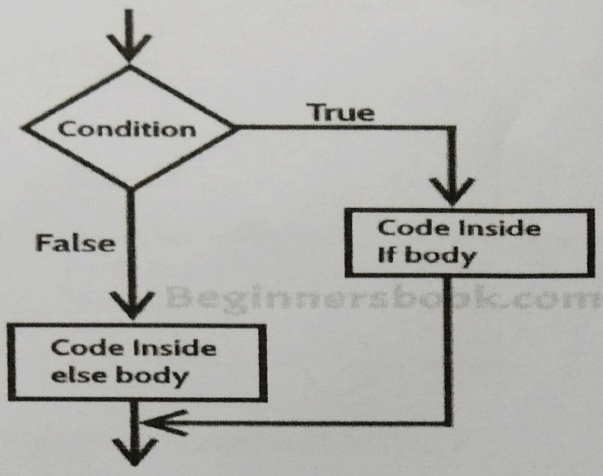
2.nested if

if-else:

```
Syntax:
if (test expression)
  {
    Block of statements;
  }
else
  {
```

```
    Block of statements;
  }
```

flow chart:



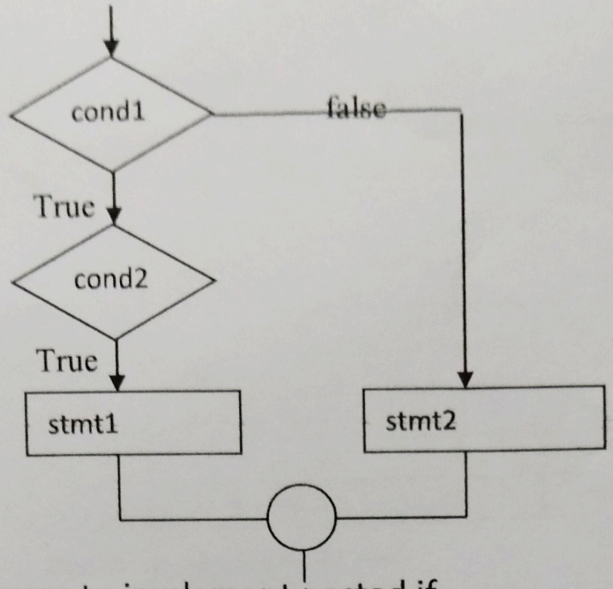
If the test expression is evaluated to true,
statements inside the body of if are executed.
statements inside the body of else are skipped from execution.

If the test expression is evaluated to false,
statements inside the body of else are executed
statements inside the body of if are skipped from execution.
Nested if:
placing if condition inside another if s called nested if

syntax:

```
if(cond1)
{
  if(cond2)
    {
       stmt1;
     }
}
else
{
   stmt2;
}
```

flow chart:



example: c program to implement nested if

```
#include<stdio.h>
void main()
{
   int p,q,r,s;
   printf ("enter p,q,r,s values");
  scanf("%d%d%d%d",&p,&q,&r,&s);
  if(r>0&&s>0)
    {
        if(p%2==0)
         {
            if(q>r&&s>p)
              {
                if(r+s>p+q)
                  {
                      printf("correct values");
                  }
              }
         }
    }
 else
   {

     printf("wrong values");
   }
 }
```

10 (a) Explain about exact size integers with suitable example?

Sol:

	Type	Description
Signed	int8_t	8-bit signed integer
	int16_t	16-bit signed integer
	int32_t	32-bit signed integer
	int64_t	64-bit signed integer
Unsigned	uint8_t	8-bit unsigned integer
	uint16_t	16-bit unsigned integer
	uint32_t	32-bit unsigned integer
	uint64_t	64-bit unsigned integer

Table 14-1 Fixed-size Integer Types

10.b) Write a C program that receives two integer numbers and one arithmetic operator as input and performs the arithmetic operation between two numbers and displays the result.

```
#include <stdio.h>
void main()
{
int num1,num2;
float result;
char ch;
 printf("Enter first number: ");
scanf("%d",&num1);
printf("Enter second number: ");
scanf("%d",&num2);
printf("Choose operation to perform (+,-,*,/,%): ");
scanf(" %c",&ch);
switch(ch)
{
case '+':
        result=num1+num2;
        break;
case '-':
        result=num1-num2;
        break;
case '*':
        result=num1*num2;
        break;
case '/':
        result=(float)num1/(float)num2;
        break;
case '%':
          result=num1%num2;
          break;
default:
          printf("Invalid operation.\n");
}
printf("Result: %d %c %d = %f\n",num1,ch,num2,result);

}
```

OR