Mts. DDB & KNG BBA BCA COLLEGE AMRAPUR

Subject code: CS – 02

BCA SEMESTER 1

Subject Name: Problem Solving Methodologies and Programming in C

UNIT 2 CONTROL STRUCTURE

SECTION A

1. **The continue statement cannot be used with ……**
   Continue is used to skip the statements and cannot be used with switch.

2. **Which keyword can be used for coming out of recursion?**
   Return is used for coming out of recursion.

3. **goto can be used to jump from main to within a function?**
   No
   goto statement in C programming provides an unconditional jump from the 'goto' to a labeled statement in the same function.

4. **do-while loop terminates when conditional expression returns?**
   Zero indicate False which terminate the loop.

5. **What will be the output of the following piece of code?**

   ```
   for(i = 0; i<10; i++);
   printf("%d", i);
   ```

   Output : 10

6. **What will be the output of the following code?**

   ```
   #include
   void main()
   {
      int s=0;
               while(s++<10)
    {
       if(s<4 && s<9)
       continue;
       printf("%dt", s);
    }
   ```

```
}
```

Output: 4 5 6 7 8 9 10

7. **What is the output of this program?**

```
#include<stdio.h>

void main()

{

  int j = -5;

  for(;j;printf("%d ", j++));

}
```

Output: -5 to -1
for loop can be initialized outside of the loop. Since until -1 value of i remain a non-zero value and hence the condition is true up to -1.

8. **Which loop used, when we not check first time condition?**
Do…..while

9. **In switch…….case, case label must be end with** colon :

10. **Determine Output:**

```
void main()
{
     int i;
     printf("%d", scanf("%d", &i)); // value 10 is given as input here
}
```

Output: 1

11. **What is the output of this program?**

```
#include <stdio.h>

void main()

{

    int a=10;

    if(a=5)

        printf("YES");

    else

        printf("NO");

}
```

Output : YES

12. **Which loop is known as post-test loop?**
do…while

13. **Which number is known as Armstrong number?**
Sum of its digit cube and original number are same.

SECTION B

1. **Explain continue statement with example.**

   The continue statement is used to bring the program control to the beginning of the loop. The continue statement skips some lines of code inside the loop and continues with the next iteration.

   It is mainly used for a condition so that we can skip some code for a particular condition.

   Syntax:

```
//loop statements
continue;
//some lines of the code which is to be skipped
```

   Example:

```
#include<stdio.h>
int main(){
int i=1;//initializing a local variable
//starting a loop from 1 to 10
for(i=1;i<=10;i++){
if(i==5){//if value of i is equal to 5, it will continue the loop
continue;
}
printf("%d \n",i);
}//end of for loop
return 0;
}
```

2. **Explain goto statement with example.**

   The goto statement is known as jump statement in C. goto is used to transfer the program control to a predefined label. The goto statement can be used to repeat some part of the code for a particular condition.

   Syntax:

```
label:
//some part of the code;
goto label;
```

   Example :

```
#include <stdio.h>
int main()
{
 int num,i=1;
          printf("Enter the number whose table you want to print?");
           scanf("%d",&num);
            table:
            printf("%d x %d = %d\n",num,i,num*i);
            i++;
            if (i<=10)
            goto table;
}
```

   Output:

```
Enter the number whose table you want to print?10
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
10 x 9 = 90
10 x 10 = 100
```

3. **What is switch statement?**

   A switch statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each switch case.

4. **List out various if statement in C.**

   - If statement
   - If-else statement
   - If else-if ladder
   - Nested if

5. **What is a nested loop?**

   A loop that runs within another loop is referred to as a **nested loop.** The first loop is called the Outer Loop and the inside loop is called the Inner Loop. The inner loop executes the number of times defined in an outer loop.

6. **What are the valid places to have keyword "Break"?**

   The purpose of the Break keyword is to bring the control out of the code block which is executing. It can appear only in looping or switch statements

7. **Write rules for switch statement.**

   - An expression must always execute to a result.
   - Case labels must be constants and unique.
   - Case labels must end with a colon (:).
   - A break keyword must be present in each case.
   - There can be only one default label.

8. **Write example of If …else statement.**

   Example:

```
#include<stdio.h>
int main()
{
int num=19;
if(num<10)
{
        printf("The value is less than 10");
}
else
{
        printf("The value is greater than 10");
}
return 0;
}
```

SECTION C

1. **What are macros? What are its advantages and disadvantages?**
   Macros are abbreviations for lengthy and frequently used statements. When a macro is called the entire code is substituted by a single line though the macro definition is of several lines. The advantage of macro is that it reduces the time taken for control transfer as in case of function. The disadvantage of it is here the entire code is substituted so the program becomes lengthy if a macro is called several times.

2. **Explain if..else statement with example.**
   The general form of a simple if...else statement is,

```
if(expression)

{

  statement block1;

}

else

{

 statement block2;

}
```

If the expression is true, the statement-block1 is executed, else statement-block1 is skipped and statement-block2 is executed.

Example:

```
#include <stdio.h>

void main( )

{

  int x, y;

  x = 15;
```

```
     y = 18;

     if (x > y )

     {

      printf("x is greater than y");

     }

     else

     {

printf("y is greater than x");

     }

     }
```

3. **Write a C program to print Fibonacci series.**

```
#include<stdio.h>
int main()
{
 int n1=0,n2=1,n3,i,number;
 printf("Enter the number of elements:");
 scanf("%d",&number);
 printf("\n%d %d",n1,n2);//printing 0 and 1
 for(i=2;i<number;++i)//loop starts from 2 because 0 and 1 are already print
ed
 {
  n3=n1+n2;
  printf(" %d",n3);
  n1=n2;
  n2=n3;
 }
  return 0;
 }
```

Output :

```
Enter the number of elements:15
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377
```

4. **Write a C program to print following series.**

```
1
12
123
1234
12345
```

Program:

```c
#include <stdio.h>

int main()
{
    int i, j, N;

    printf("Enter N: ");
    scanf("%d", &N);

    for(i=1; i<=N; i++)
    {
        // Logic to print numbers
        for(j=1; j<=i; j++)
        {
            printf("%d", j);
        }

        printf("\n");
    }

    return 0;
}
```

5. Explain Switch case statement with example.

- **Switch statement** tests the value of a variable and compares it with multiple cases. Once the case match is found, a block of statements associated with that particular case is executed.
- Each case in a block of a switch has a different name/number which is referred to as an identifier. The value provided by the user is compared with all the cases inside the switch block until the match is found.
- If a case match is NOT found, then the default statement is executed, and the control goes out of the switch block.

Syntax:

```
     switch( expression )

{
case value-1:
                    Block-1;
                    Break;
case value-2:
                    Block-2;
                    Break;
case value-n:
                    Block-n;
                    Break;
default:
                    Block-1;
                    Break;
}
Statement-x;
```

6. **Why do we need a Switch case?**

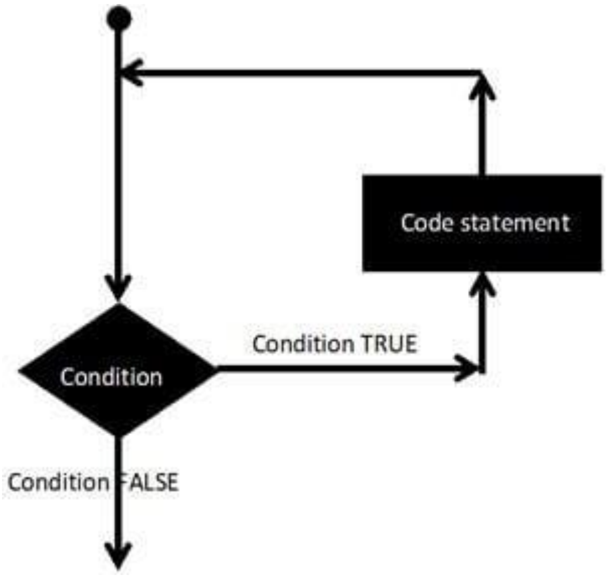
   There is one potential problem with the if-else statement which is the complexity of the program increases whenever the number of alternative path increases. If you use multiple if-else constructs in the program, a program might become difficult to read and comprehend. Sometimes it may even confuse the developer who himself wrote the program.The solution to this problem is the switch statement.

SECTION D

1. **What are the general description of loop statements and available loop types in C?**

   A statement that allows the execution of statements or groups of statements in a repeated way is defined as a loop.

   The following diagram explains a general form of a loop.



There are 4 types of loop statements in C.

- While loop
- For Loop
- Do…While Loop
- Nested Loop

2. Write a c program for check if number is odd or even.

```
#include<stdio.h>
int main()
{
  int num;

   printf("Enter an integer: ");
   scanf("%d",&num);

   if ( num%2 == 0 )
    printf("%d is an even number", num);
  else
   printf("%d is an odd number", num);

  return 0;
}
```

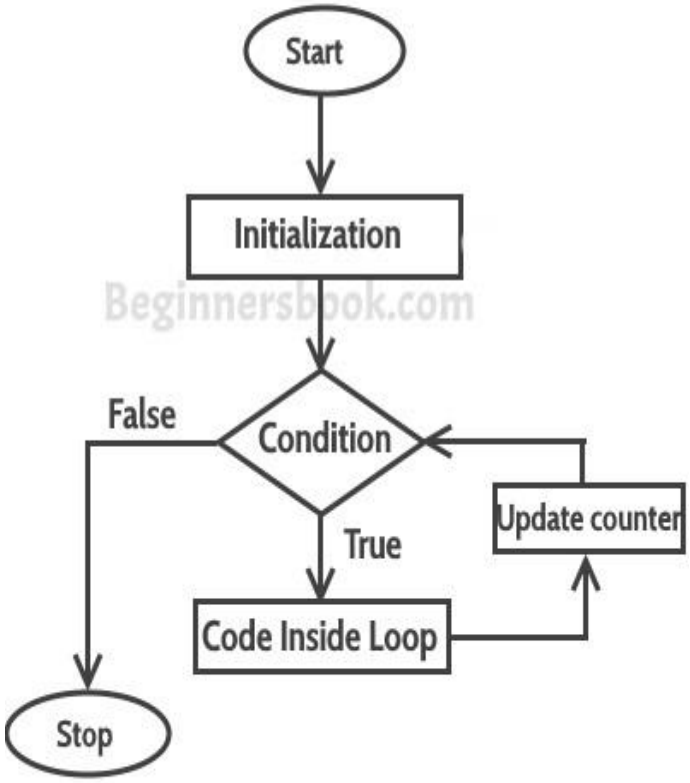
3. Explain For loop with Example.

A loop is used for executing a block of statements repeatedly until a given condition returns false.

For loop

This is one of the most frequently used loop in C programming.
Syntax of for loop:

```
   for (initialization; condition test; increment or decrement)
   {
//Statements to be executed repeatedly
   }
```



Step 1: First initialization happens and the counter variable gets initialized.

Step 2: In the second step the condition is checked, where the counter variable is tested for the given condition, if the condition returns true then the C statements inside the body of for loop gets executed, if the condition returns false then the for loop gets terminated and the control comes out of the loop.

Step 3: After successful execution of statements inside the body of loop, the counter variable is incremented or decremented, depending on the operation (++ or –).

Example:-

```
#include <stdio.h>
int main()
{
  int i;
  for (i=1; i<=3; i++)
```

```
   {
      printf("%d\n", i);
   }
   return 0;
}
```

4. **Explain types of loop in detail.**

   There are two types of loop in C language.

   1 **Entry controlled loop**

   In this loop, during the execution process of loop it first checks the condition and if the condition is true then and then it will execute body of the loop otherwise control will directly jump to next statement after the body loop. That's why this type of loop is known as Entry Controlled loop.

   For Example: for and while loop structure are an example of Entry Controlled loop.

   For loop : A loop is used for executing a block of statements repeatedly until a given condition returns false.

Syntax of for loop:

```
for (initialization; condition test; increment or decrement)
{
  //Statements to be executed repeatedly
}
```

While loop : A loop is used for executing a block of statements repeatedly until a given condition returns false.

Syntax of while loop:

```
while (condition test)
{
  //Statements to be executed repeatedly
 // Increment (++) or Decrement (--) Operation
}
```

2 **Exit controlled loop:** This type of loop first execute its body statement and at the end of body it check the condition. If condition becomes true then it will repeat same process again or execute loop body again. But if condition becomes false then it will terminate loop process and control will move on next statement.
This type of loop execute its statement at least one time and at the end of loop check the condition, that's t why this type of loop called as Exit Controlled loop.

Example: do…while loop is an example of exit controlled loop.

5. **Explain nested if….else statement with example.**
The if statement may itself contain another if statement is called as nested if…else statement. Whenever we required to check condition inside other condition , such as situation is known as nested condition.

The general form of this statement is given as…

```
if(condition)
 {
   //Nested if else inside the body of "if"
   if(condition2) {
    //Statements inside the body of nested "if"
 }
  else {
   //Statements inside the body of nested "else"
}
}
else {
 //Statements inside the body of "else"
}
```

Example of nested if..else

```
#include <stdio.h>
int main()
{
  int var1, var2;
  printf("Input the value of var1:");
  scanf("%d", &var1);
  printf("Input the value of var2:");
  scanf("%d",&var2);
  if (var1 != var2)
  {
         printf("var1 is not equal to var2\n");
         //Nested if else
         if (var1 > var2)
         {
                  printf("var1 is greater than var2\n");
         }
         else
         {
                  printf("var2 is greater than var1\n");
         }
  }
  else
  {
         printf("var1 is equal to var2\n");
  }
  return 0;
}
```