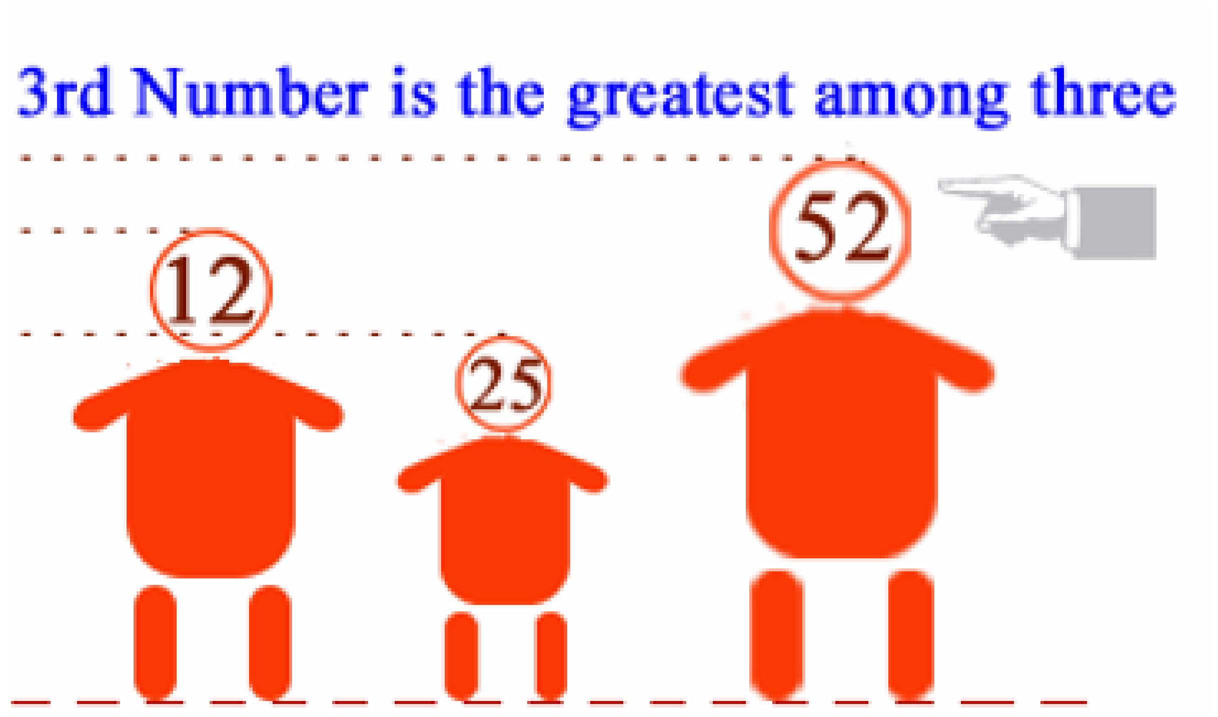
Fundamentals of Programming CCS1063/CSE1062

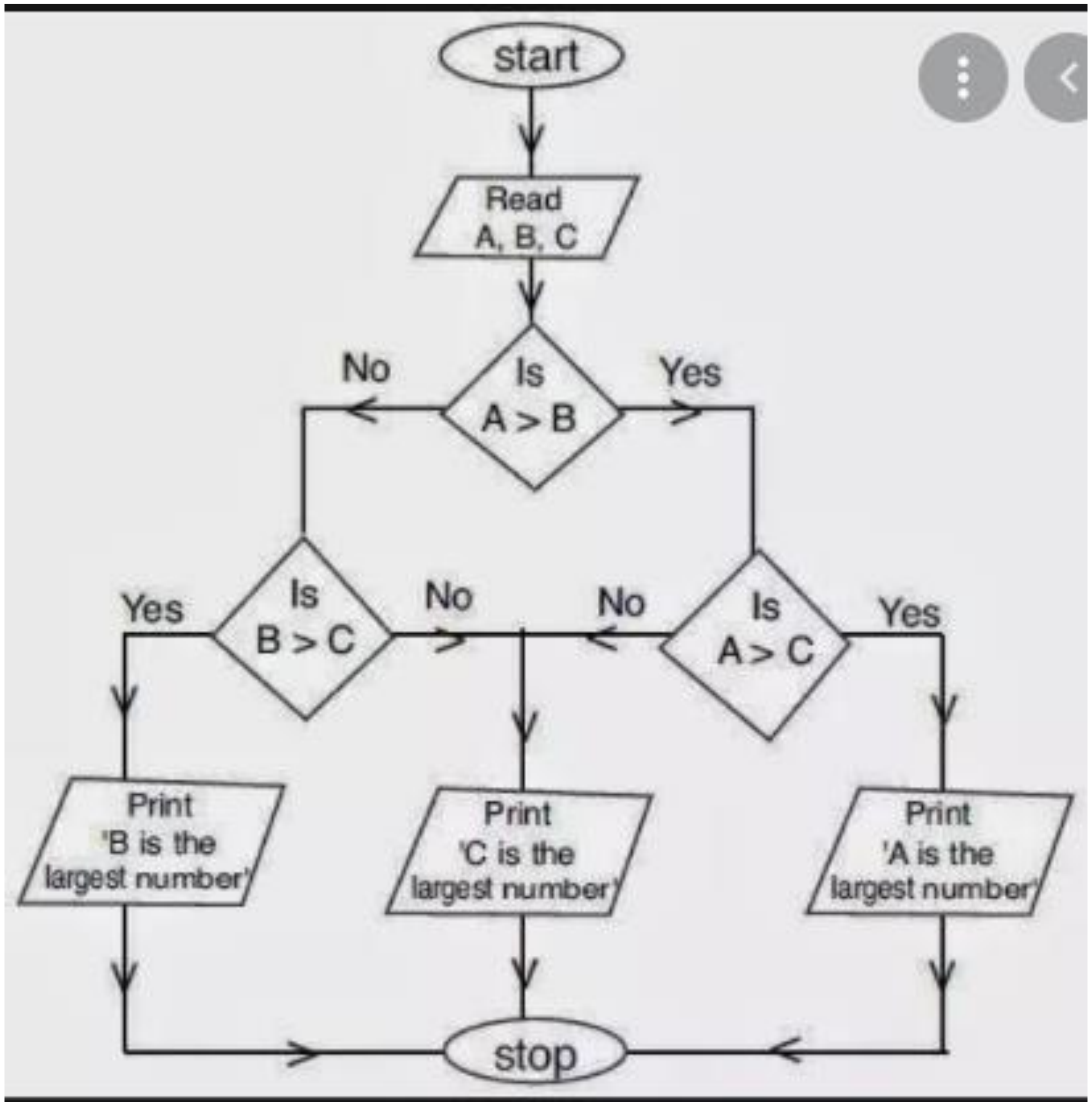
Lecture 6 –Operators Part 2 , Conditions and Loops

Professor Noel Fernando

Question

- Write a C program to find the largest of three numbers.




start
Read A, B, C
Is A > B
No
Yes
Is B > C
Yes
No
No
Is A > C
Yes
Print 'B is the largest number'
Print 'C is the largest number'
Print 'A is the largest number'
stop

Answer

```
#include <stdio.h>
void main()
{
    int num1, num2, num3;

    printf("Input the values of three numbers : ");
    scanf("%d %d %d", &num1, &num2, &num3);
    printf("1st Number = %d,\t2nd Number = %d,\t3rd Number =
%d\n", num1, num2, num3);
    if (num1 > num2)
    {
        if (num1 > num3)
        {
printf("The 1st Number is the greatest among three. \n");
        }
        else
        {
            printf("The 3rd Number is the greatest among three. \n");
        }
    }
    else if (num2 > num3)
        printf("The 2nd Number is the greatest among three \n");
    else
        printf("The 3rd Number is the greatest among three \n");
}
```

Alternative Algorithm

Control Statements Cond….

- Exercise: Draw a flow chart, Write pseudo code to determine the grades
- Hint : marks >=90 “A GRADE”

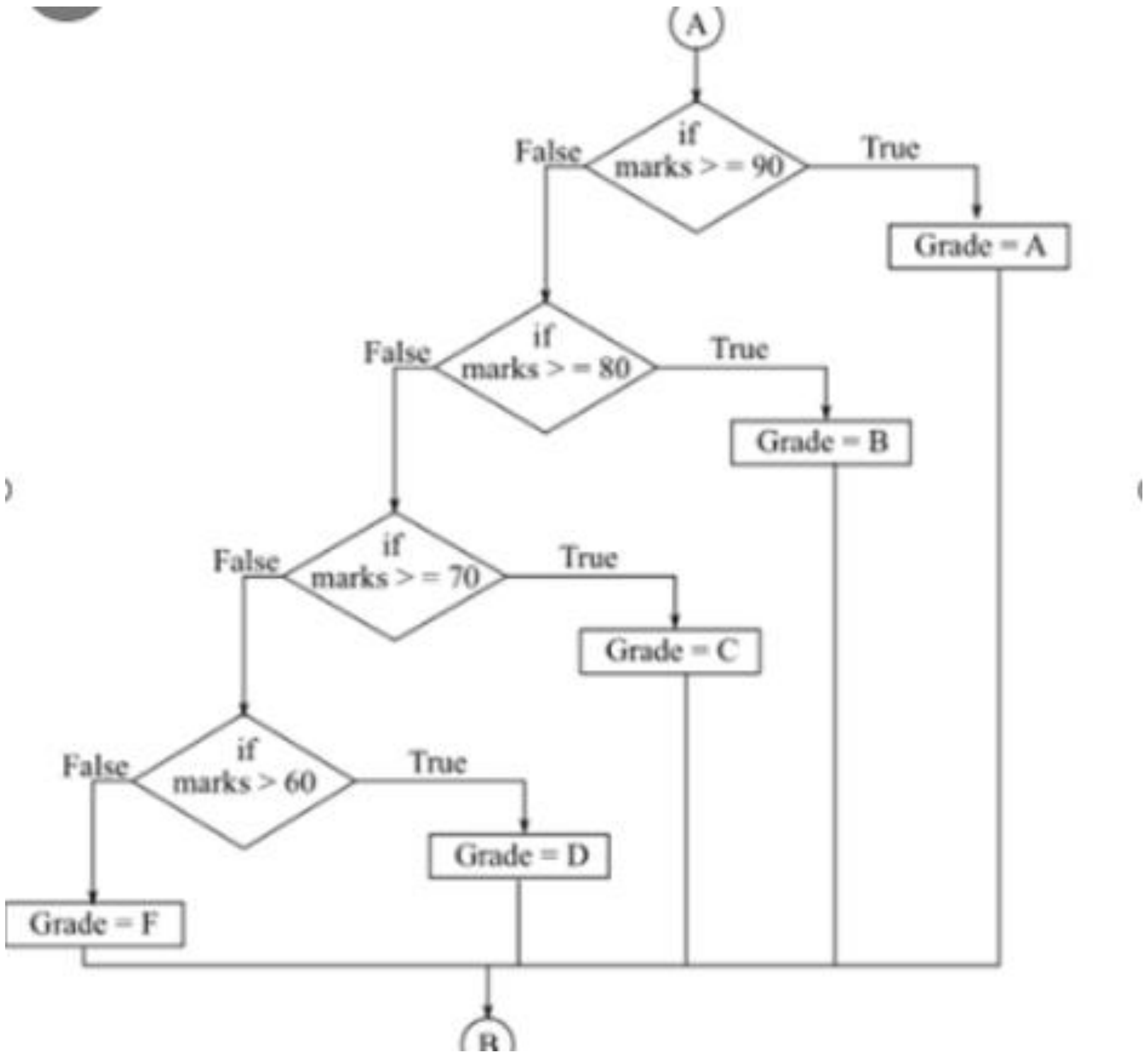
  marks >=80 and =< 89 “B GRADE”

  marks >=7 0and =< 79 “C GRADE”

  marks >=60 and =< 69 “D” GRADE”

  Otherwise “ F GRADE”

Flow Chart


A
if marks > = 90
False
True
Grade = A
if marks > = 80
False
True
Grade = B
if marks > = 70
False
True
Grade = C
if marks > 60
False
True
Grade = D
Grade = F
B

Pseudocode Algorithm

```
Read marks
if marks >=90 then
    print( "A Grade")
Else if  (marks >=80)  then
    print (" B Grade")
Else if  (marks>= 70 ) then
     print ("C Grade")
Else if  marks >=60 then
     print (“D Grade")
else
     print ("F Grade ")
endif
```

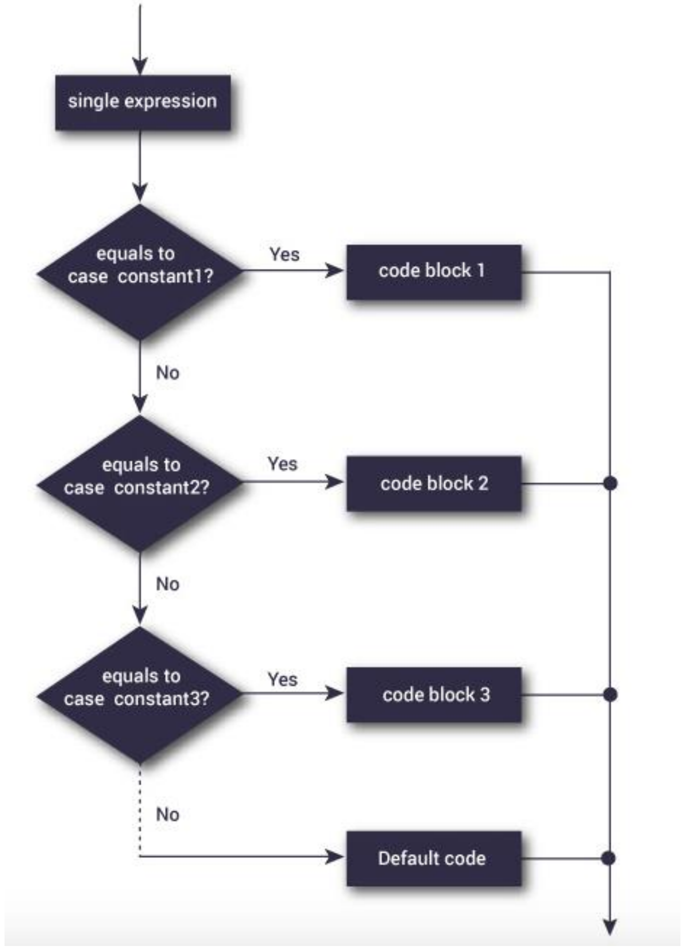
Example: Grades

- Write a program to give the grade when you enter your mark of a subject.
- The grade of the mark is defined as follows.
    - 00 – 24: E
    - 25 – 34: D
    - 35 – 49: C
    - 50 – 69: B
    - 70 – 100: A

Example: Grades

```
int main() {

    int mark;
    char grade;

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

    if(mark>69) {
        grade = 'A';
    }
    else if(mark>49) {
            grade = 'B';
    }
    else if(mark>34) {
            grade = 'C';
    }
    else if(mark>24) {
        grade = 'D';
    }
    else {
        grade = 'E';
    }
    printf("your grade is %c", grade);
}
```

case Statement



- The if..else..if ladder allows you to execute a block code among many alternatives.
- If you are checking on the value of a single variable in if...else...if, it is better to use switch statement.
- The switch statement is often faster than multiple if...else.

case Statement

- The syntax of switch statement in c language

```
switch(expression){
case value1:
 //code to be executed;
 break;  //optional
case value2:
 //code to be executed;
 break;  //optional
......
   default:
 code to be executed if all cases are not matched;
}
```

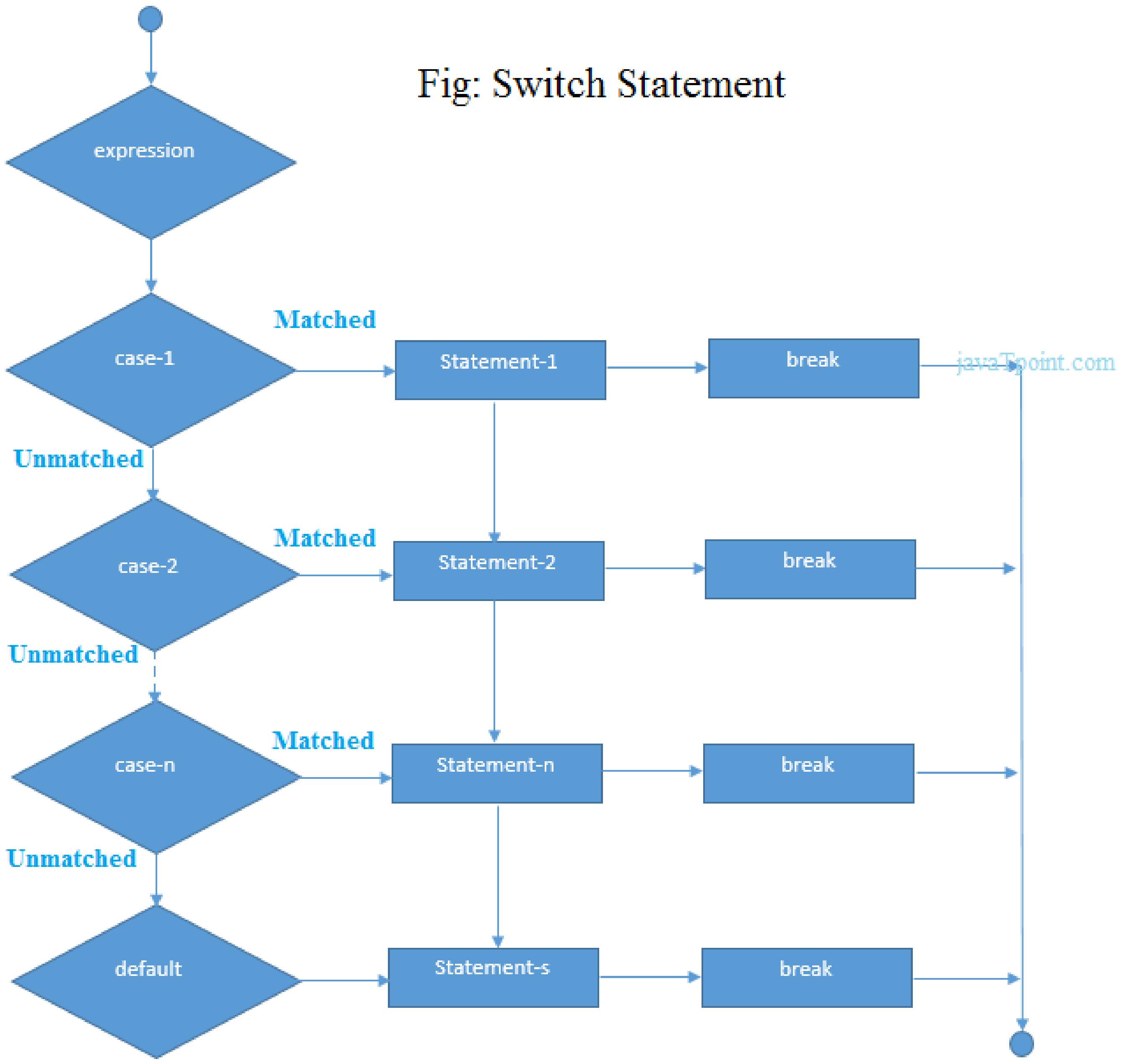


Fig: Switch Statement

Question: Write a C program to find maximum between two numbers using switch case

Example

Input
Input first number: 12 Input second number: 40

Output
Maximum: 40

Answer: Write a C program to find maximum between two numbers using switch case

```
#include <stdio.h>
int main()
{
    int num1, num2;
    printf("Enter two numbers to find maximum: ");
    scanf("%d%d", &num1, &num2);
    /* Expression (num1 > num2) will return either 0 or 1 */
```

Answer: Write a C program to find maximum between two numbers using switch case

```
 switch(num1 > num2)
   {
       /* If condition (num1>num2) is false */
       case 0:
           printf("%d is maximum", num2);
           break;
       /* If condition (num1>num2) is true */
       case 1:
           printf("%d is maximum", num1);
           break;
   }
   return 0;
}
```

Example:

```c
int main () {

    char grade;

    printf("Enter your grade: ");
    scanf("%c", &grade);

    switch(grade) {
        case 'A' :
            printf("Excellent!\n" );
            break;
        case 'B' :
        case 'C' :
            printf("Well done\n" );
            break;
        case 'D' :
            printf("You passed\n" );
            break;
        case 'F' :
            printf("Better try again\n" );
            break;
        default :
            printf("Invalid grade\n" );
    }

    printf("Your grade is  %c\n", grade);

    return 0;
}
```

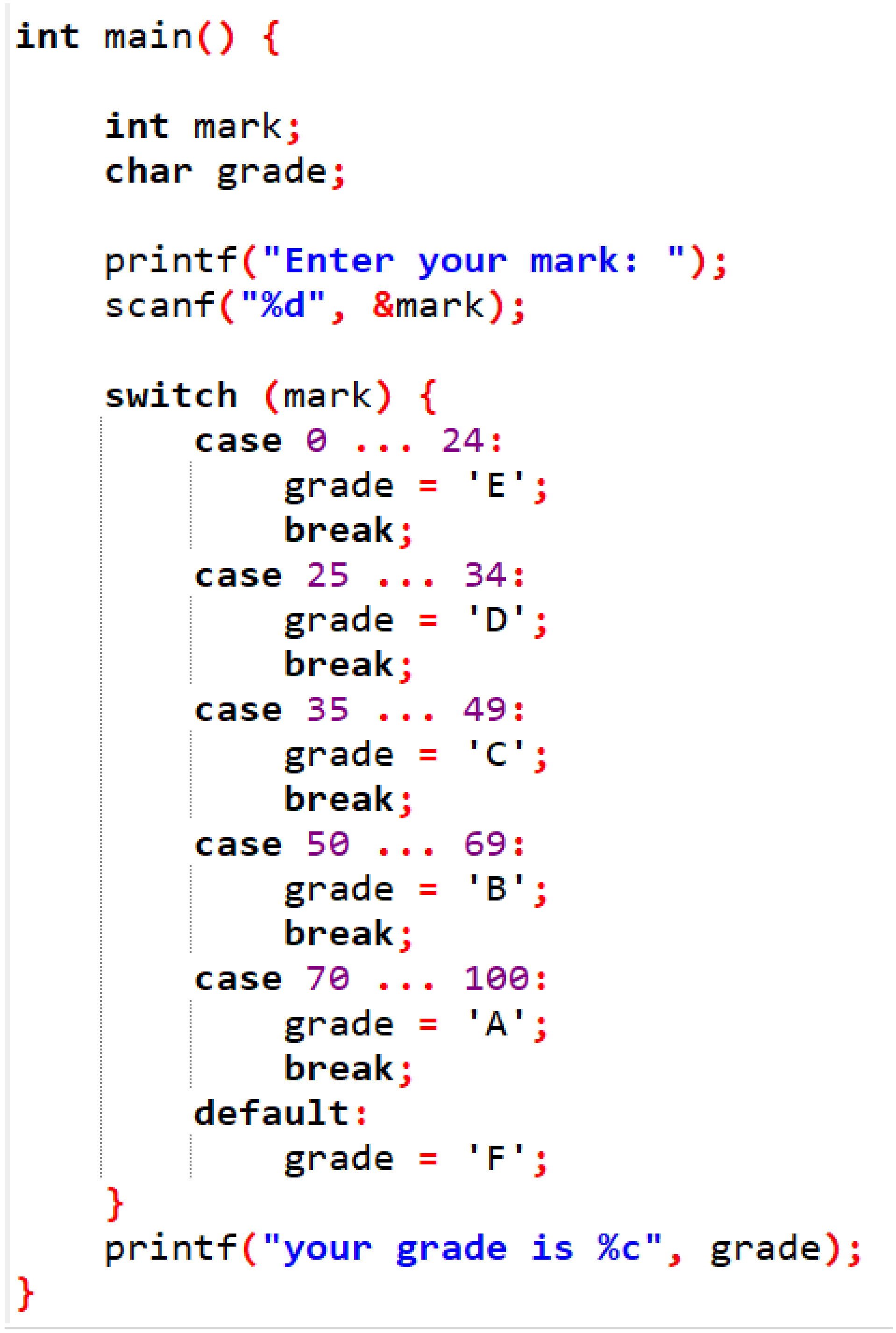
Rules of case statement

1. The **expression** used in a **switch** statement must have an integral or enumerated type, or be of a class type in which the class has a single conversion function to an integral or enumerated type.
2. You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.
3. The **constant-expression** for a case must be the same data type as the variable in the switch, and it must be a constant or a literal.
4. When the variable being switched on is equal to a case, the statements following that case will execute until a **break** statement is reached.
5. When a **break** statement is reached, the switch terminates, and the flow of control jumps to the next line following the switch statement.
6. Not every case needs to contain a **break**. If no **break** appears, the flow of control will fall through to subsequent cases until a break is reached.
7. A **switch** statement can have an optional **default** case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases is true. No **break** is needed in the default case.

Question

- Write a C code print the grade according to the following marks ranges
- Hint: you may use switch statements
- 0-24 →E
- 25-34 →D
- 35-49 →C
- 50-69 →B
- 70-100→A
- Default :F

Example: Grades



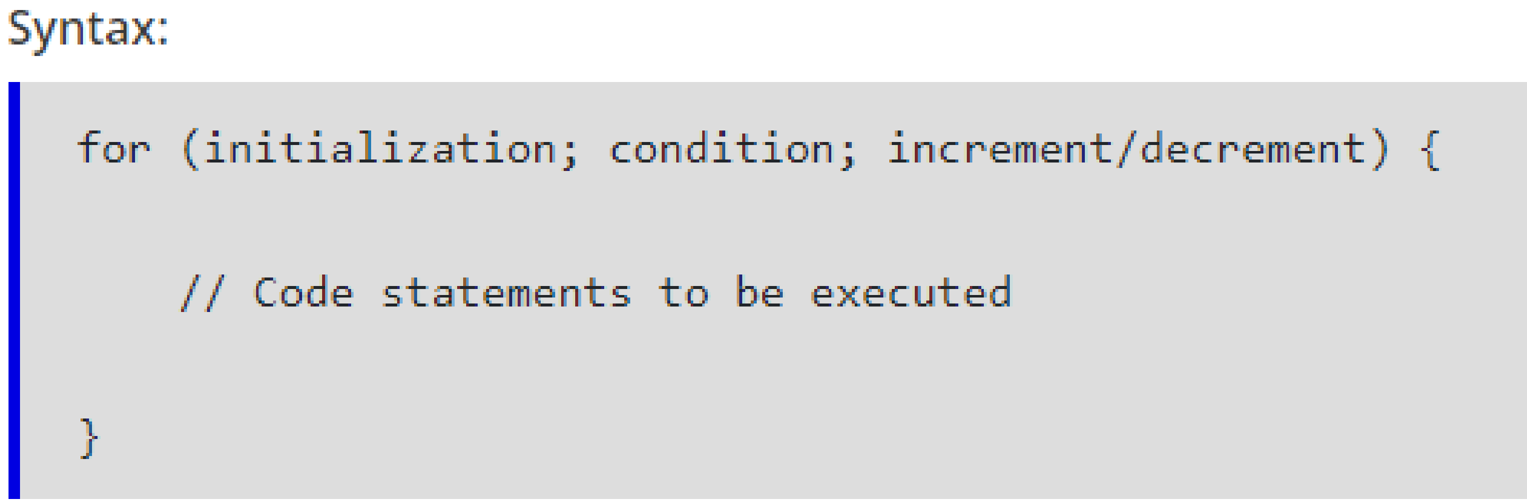
```c
int main() {

    int mark;
    char grade;

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

    switch (mark) {
        case 0 ... 24:
            grade = 'E';
            break;
        case 25 ... 34:
            grade = 'D';
            break;
        case 35 ... 49:
            grade = 'C';
            break;
        case 50 ... 69:
            grade = 'B';
            break;
        case 70 ... 100:
            grade = 'A';
            break;
        default:
            grade = 'F';
    }
    printf("your grade is %c", grade);
}
```

Loops

- Loops are used in programming to repeat a specific block of code.
- There are three loops in C programming:
  1. for loop
  2. while loop
  3. do...while loop

For loop

- It also executes the code until condition is false.
- In this three parameters are given that is:
- Initialization
- Condition
- Increment/Decrement



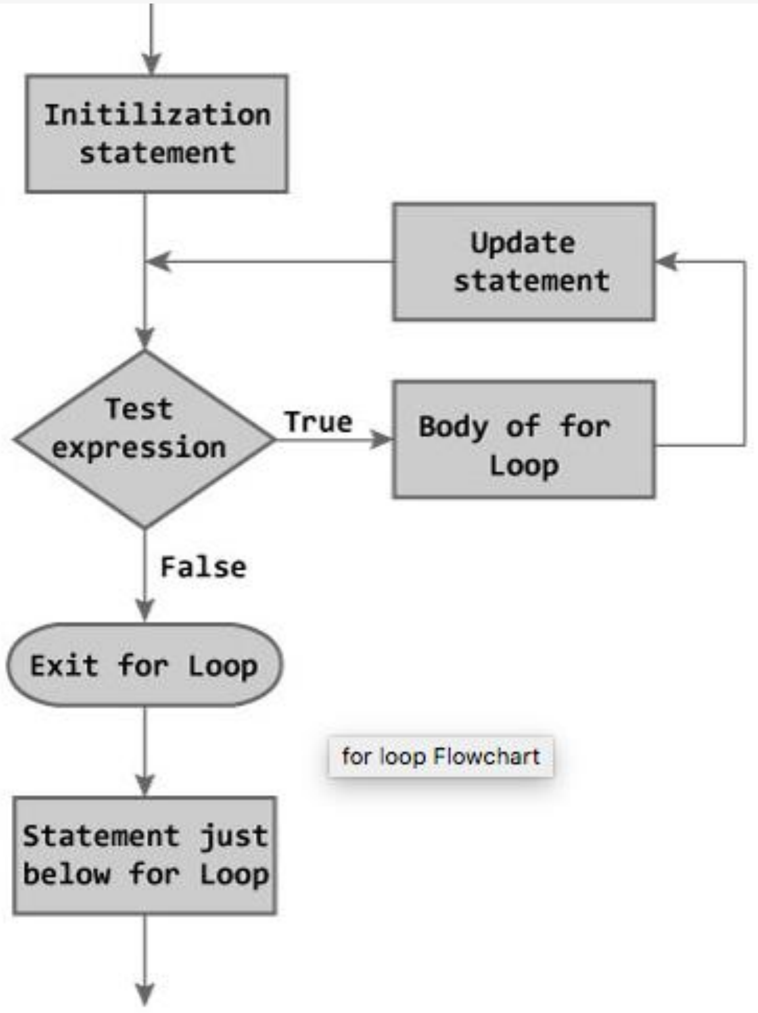
Syntax:

```
for (initialization; condition; increment/decrement) {

    // Code statements to be executed

}
```

for Loop

```
for (initializationStatement; testExpression; updateStatement)
{
        // codes
}
```

1. The initialization statement is executed only once.
2. Then, the test expression is evaluated. If the test expression is false (0), the loop is terminated.
3. If the test expression is true (nonzero), codes inside the body of the loop is executed.
4. Then, the update statement is executed and update the variable.
5. This process repeats **until the test expression is false**.
6. The for loop is commonly used **when the number of iterations is known**.



for loop Flowchart

Example

```
#include<stdio.h>

void main()

{

int i;

for( i = 20; i < 25; i++) {

printf ("%d " , i);

}

}
```

Output:

```
20 21 22 23 24
```

Example: Write a C program to Find Factors

- Example :

- If number is 20 ,
- then factors of 20 are :1,2,4,5,10,20

Write a C program to Find Factors

```
#include <stdio.h>
int main() {
    int num, i;
    printf("Enter a positive integer: ");
    scanf("%d", &num);
    printf("Factors of %d are: ", num);
      for (i = 1; i <= num; ++i) {
         if (num % i == 0) {
            printf("%d ", i);
         }
       }
       return 0;
}
```

Nested for Loops

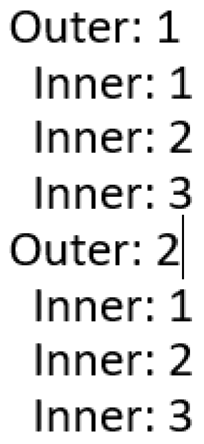
- It is also possible to place a loop inside another loop. This is called a **nested loop**.
- The "inner loop" will be executed one time for each iteration of the "outer loop":

Write down the output of the following program

```
#include <stdio.h>
int main() {
 int i, j;
  // Outer loop
 for (i = 1; i <= 2; ++i) {
  printf("Outer: %d\n", i);  // Executes 2 times

  // Inner loop
  for (j = 1; j <= 3; ++j) {
   printf(" Inner: %d\n", j);  // Executes 6 times (2 * 3)
  }
 }
  return 0;
}
```

Write down the output of the following program

```
#include <stdio.h>
int main() {
 int i, j;
  // Outer loop
 for (i = 1; i <= 2; ++i) {
  printf("Outer: %d\n", i);  // Executes 2 times


  // Inner loop
  for (j = 1; j <= 3; ++j) {
   printf(" Inner: %d\n", j);  // Executes 6 times (2 * 3)
  }
 }
  return 0;
}
```



```
Outer: 1
 Inner: 1
 Inner: 2
 Inner: 3
Outer: 2
 Inner: 1
 Inner: 2
 Inner: 3
```

While Loop

- While Loops can execute a block of code as long as a specified condition is reached.

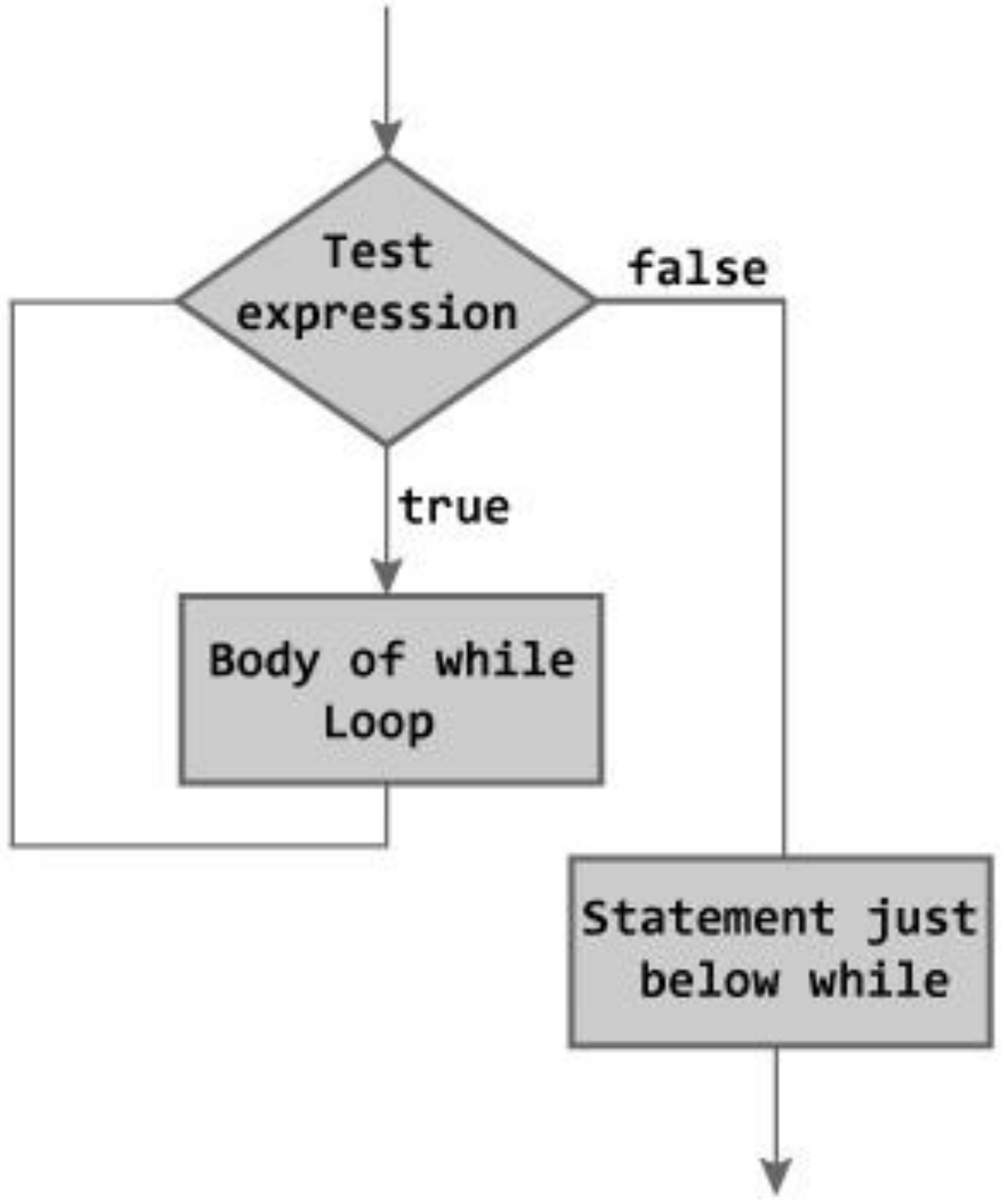
Syntax

```
while (condition) {
  // code block to be executed
}
```

- The While loops executes a block of code as long as a specified condition is True.

while Loop

- The while loop evaluates the test expression.
- If the test expression is true (nonzero), codes inside the body of while loop are executed
- The test expression is evaluated again. The process goes on until the test expression is false.
- When the test expression is false, the while loop is terminated.



Question

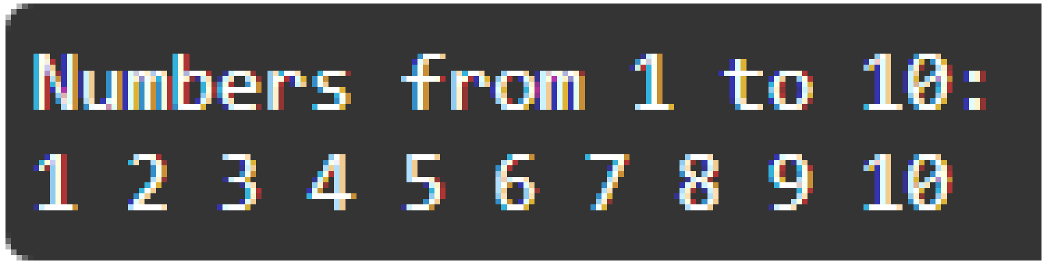
- How to print the first ten numbers with the use of a while loop?

Answer :How to print the first ten numbers with the use of a while loop

```
#include <stdio.h>
int main()
{
      int number;
      //assign initial value from where we want to print the numbers
      number =1;
      //print statement
```

Answer :How to print the first ten numbers with the use of a while loop

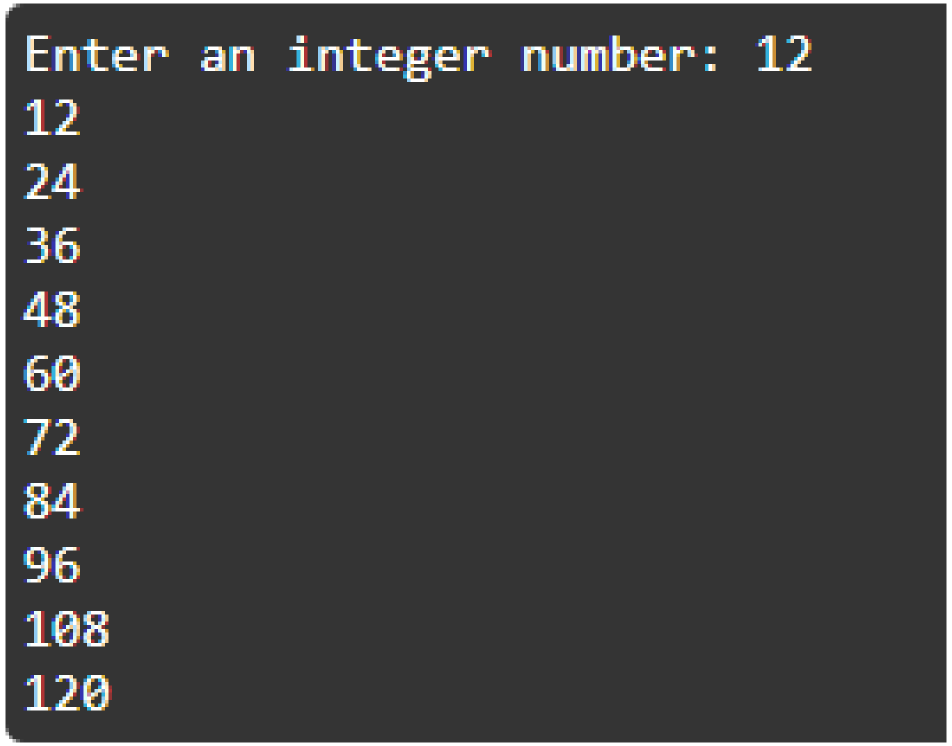
```
        printf("Numbers from 1 to 10: \n");
                while(number<=10)
        {
                //printing the numbers
                printf("%d ",number);
                //increasing loop counter by 1
                number++;
                return 0;
            }
          return 0;
}
```



```
Numbers from 1 to 10:
1 2 3 4 5 6 7 8 9 10
```

Question: Write a C code to read an integer number and print its multiplication table

- Expected output is:



```
Enter an integer number: 12
12
24
36
48
60
72
84
96
108
120
```

Question: Write a C code to read an integer number and print its multiplication table

- **Logic:**
- Read an integer number
- Take a loop counter and initialize it with 1
- Run a loop from 1 to 10
- Print the multiplication of input number and loop counter
- Increase the loop counter

Answer: Write a C code to read an integer number and print its multiplication table

```
#include <stdio.h>
int main()
{
        int num;      /*to store number*/
        int i;  /*loop counter*/
        /*Reading the number*/
        printf("Enter an integer number: ");
        scanf("%d",&num);
```

Question: Write a C code to read an integer number and print its multiplication table

```
    /*Initialising loop counter*/
    i=1;
    /*loop from 1 to 10*/
    while(i<=10){
        printf("%d\n",(num*i));
        i++; /*Increase loop counter*/
    }
    return 0;
}
```

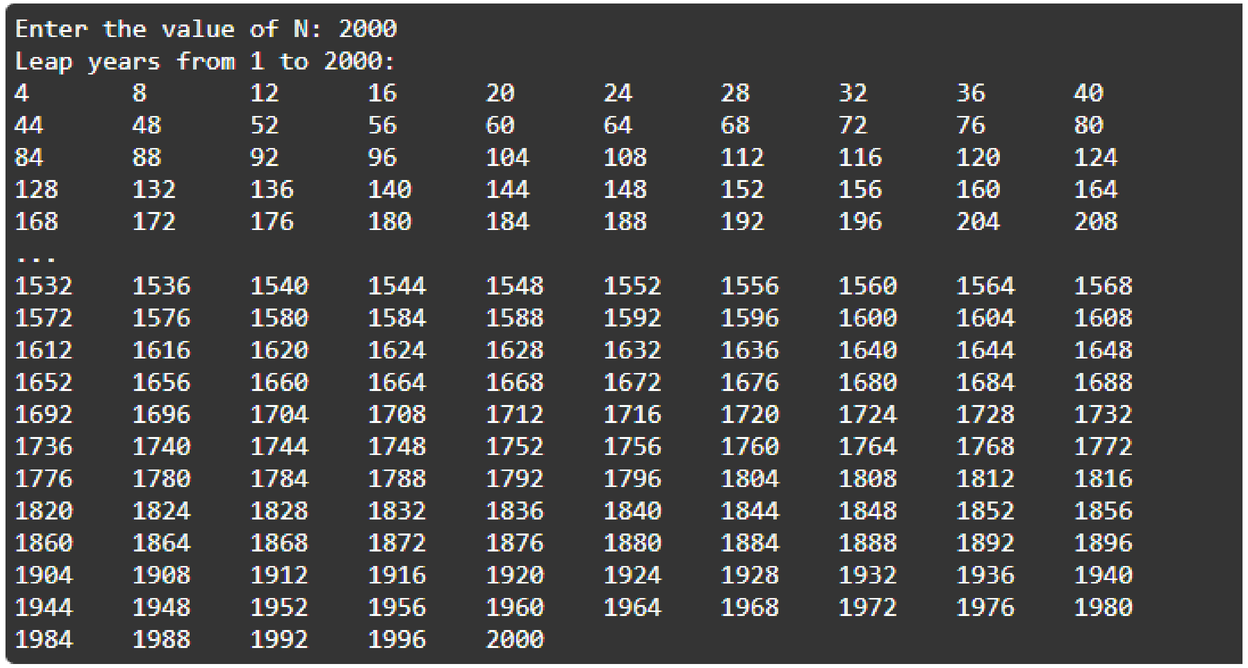
Question : write a C program to print all Leap Year from 1 to N

- This program will read value of N and **print all Leap Years from 1 to N years**.
- There are two conditions for leap year:

1- If year is divisible by 400 (for Century years),

2- If year is divisible by 4 and must not be divisible by 100 (for Non Century years).

Output



```
Enter the value of N: 2000
Leap years from 1 to 2000:
4       8       12      16      20      24      28      32      36      40
44      48      52      56      60      64      68      72      76      80
84      88      92      96      104     108     112     116     120     124
128     132     136     140     144     148     152     156     160     164
168     172     176     180     184     188     192     196     204     208
...
1532    1536    1540    1544    1548    1552    1556    1560    1564    1568
1572    1576    1580    1584    1588    1592    1596    1600    1604    1608
1612    1616    1620    1624    1628    1632    1636    1640    1644    1648
1652    1656    1660    1664    1668    1672    1676    1680    1684    1688
1692    1696    1704    1708    1712    1716    1720    1724    1728    1732
1736    1740    1744    1748    1752    1756    1760    1764    1768    1772
1776    1780    1784    1788    1792    1796    1804    1808    1812    1816
1820    1824    1828    1832    1836    1840    1844    1848    1852    1856
1860    1864    1868    1872    1876    1880    1884    1888    1892    1896
1904    1908    1912    1916    1920    1924    1928    1932    1936    1940
1944    1948    1952    1956    1960    1964    1968    1972    1976    1980
1984    1988    1992    1996    2000
```

Answer : write a C program to print all Leap Year from 1 to N

```
/*C program to print all leap years from 1 to N.*/
 #include <stdio.h>
//function to check leap year
int checkLeapYear(int year)
{
    if( (year % 400==0)||(year%4==0 && year%100!=0) )
        return 1;
    else
        return 0;
}
```

Answer : write a C program to print all Leap Year from 1 to N

```
int main()
{
   int i,n;
    printf("Enter the value of N: ");
   scanf("%d",&n);
   printf("Leap years from 1 to %d:\n",n);
   for(i=1;i<=n;i++)
   {
      if(checkLeapYear(i))
         printf("%d\t",i);
   }
      return 0;
}
```

Question : Write a C program to find factorial of a number

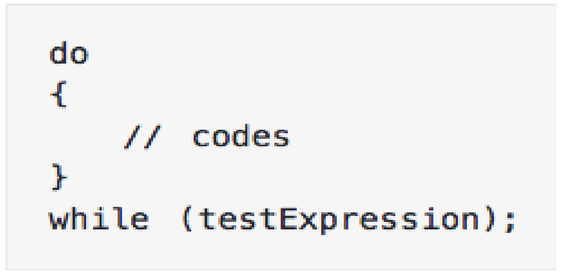
- **What is factorial**?
- ***Factorial is the product of an integer with it's all below integer till 1***.
- In mathematic representation factorial represents by ! sign.
- **For Example:**
-
  Factorial 5 is:
- 5! = 120 [That is equivalent to 5*4*3*2*1 =120]

Answer :

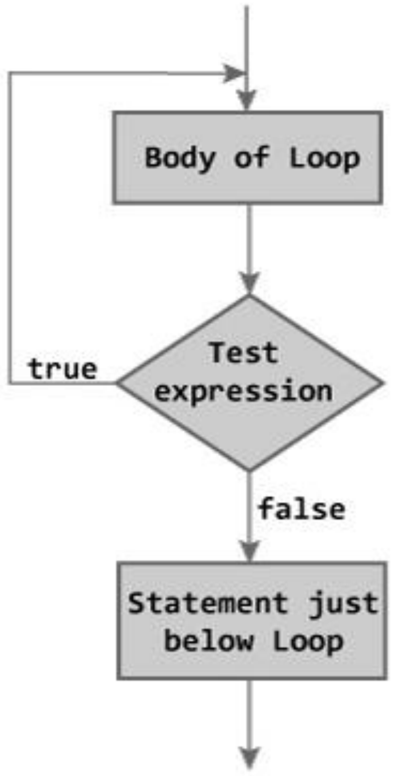
```
/*C program to find factorial of a number.*/
#include <stdio.h>
int main()
{
    int num,i;
    long int fact;
      printf("Enter an integer number: ");
    scanf("%d",&num);
     /*product of numbers from num to 1*/
    fact=1;
    for(i=num; i>=1; i--)
        fact=fact*i;
             printf("\nFactorial of %d is = %ld",num,fact);
        return 0;
}
```

do...while Loop

- The do...while loop is similar to the while loop with one important difference.
- The body of do...while loop is executed once, before checking the test expression.
- Hence, the **do...while loop is executed at least once.**



```
do
{
    //  codes
}
while  (testExpression);
```



Question : how to print 1 to 10 using do while loop in C

- **Steps are:**
- Initialize start number with 1
- Initialize target number to 10
- Enter the do while loop
- print the number
- increment the number
- put condition in while, so that if the value of num exceeds 10, then do while loop will be terminated.

Answer : how to print 1 to 10 using do while loop in C

```
#include<stdio.h>
 int main()
int num = 1;
int target = 10;
//Start do while loop
do{
printf("%d\n", num);
//increment the number by 1
 ++num;
}while (num <= target);
return 0;
}
```

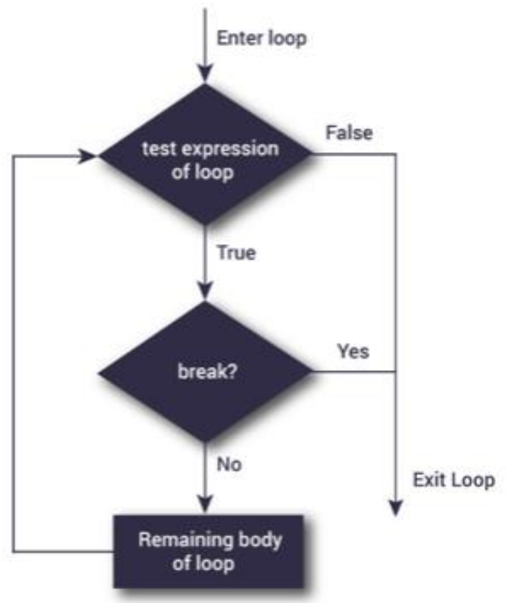
Homework: Pyramids

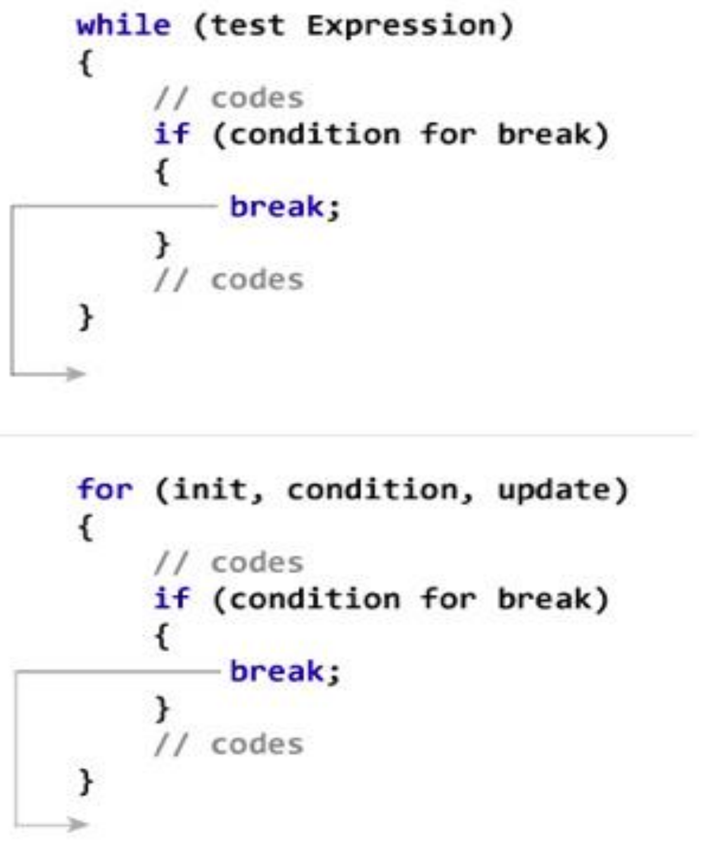
- Write a program to print the following pattern for given number of rows form the character given by the user.
  - Ex: if user enters 5 as no of rows and ‘*’ as the character, following pattern should be printed in the console.

```
*
**
***
****
*****
```

break

- The break statement terminates the loop (for, while and do...while loop) immediately when it is encountered.
- The break statement is used with decision making state





```
while (test Expression)
{
    // codes
    if (condition for break)
    {
        break;
    }
    // codes
}
```

```
for (init, condition, update)
{
    // codes
    if (condition for break)
    {
        break;
    }
    // codes
}
```

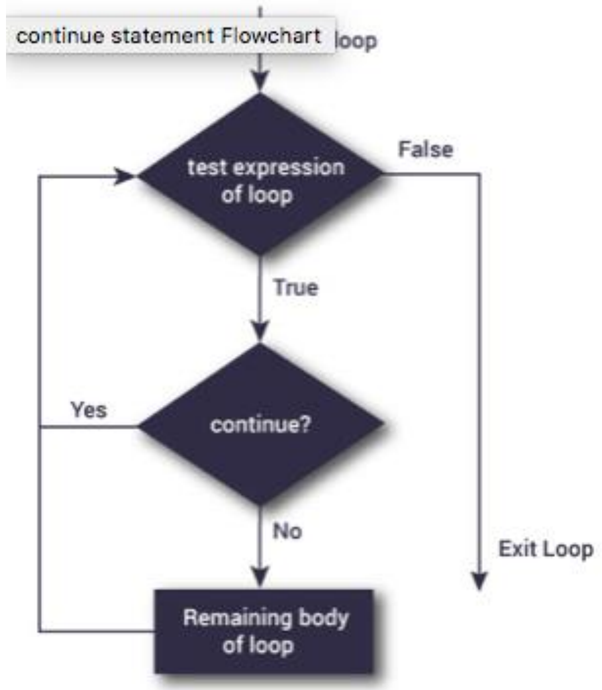
Example: break

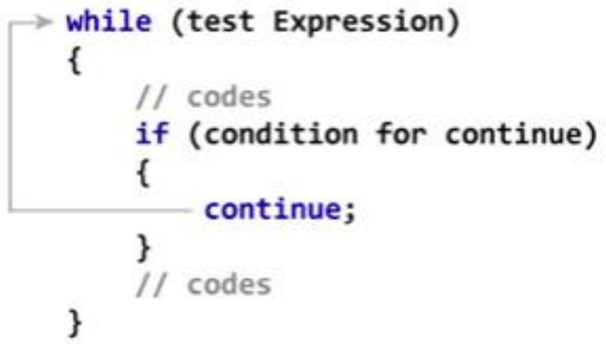
```
int main() {

    int num;

    while(1) {

        printf("\nEnter a number: ");
        scanf("%d", &num);

        if(num<0) break;
        if(num%2==0) {
            printf("%d is an even number.\n", num);
        }
        else {
            printf("%d is an odd number.\n", num);
        }
    }
    printf("Done!");
}
```

The while(1) acts as an infinite loop that runs continually until a break statement is explicitly issued

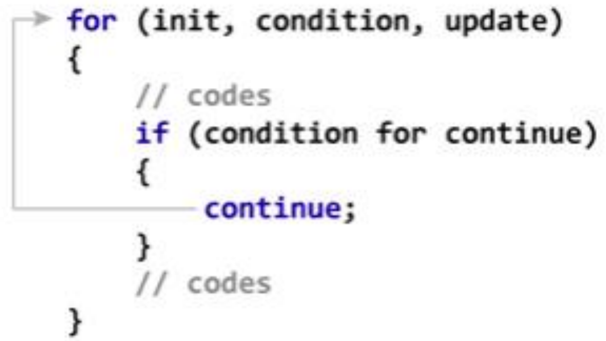
continue

- The continue statement skips some statements inside the loop.
- The continue statement is used with decision making statement such as if...else.





```
while (test Expression)
{
      // codes
      if (condition for continue)
      {
          continue;
      }
      // codes
}
```



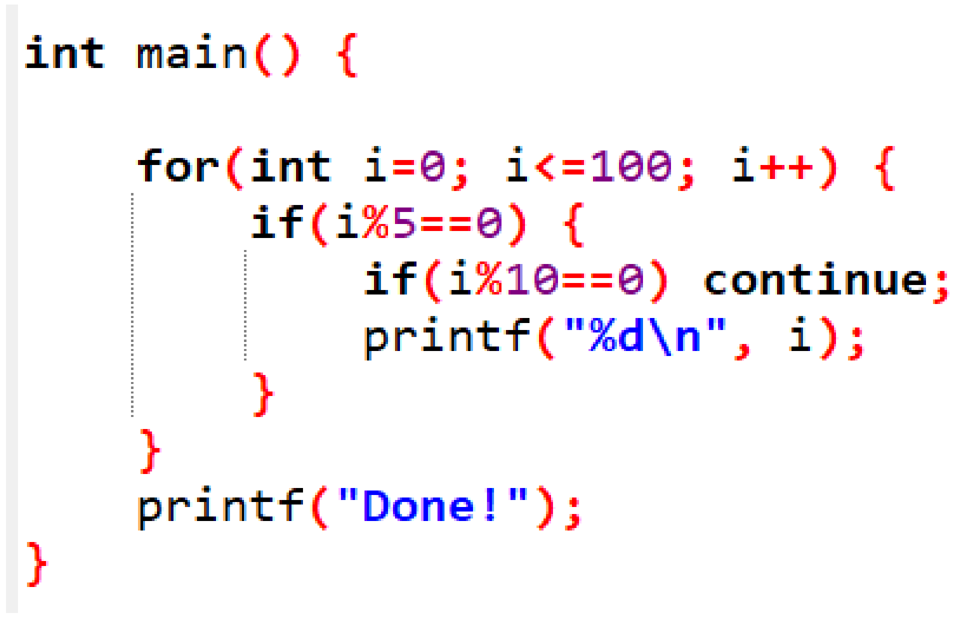
```
for (init, condition, update)
{
      // codes
      if (condition for continue)
      {
          continue;
      }
      // codes
}
```

Example: continue

- Write a program to print numbers divided by 5, but not 10 between 1-100.

Example: continue

- . Write a program to print numbers divided by 5, but not 10 between 1-100

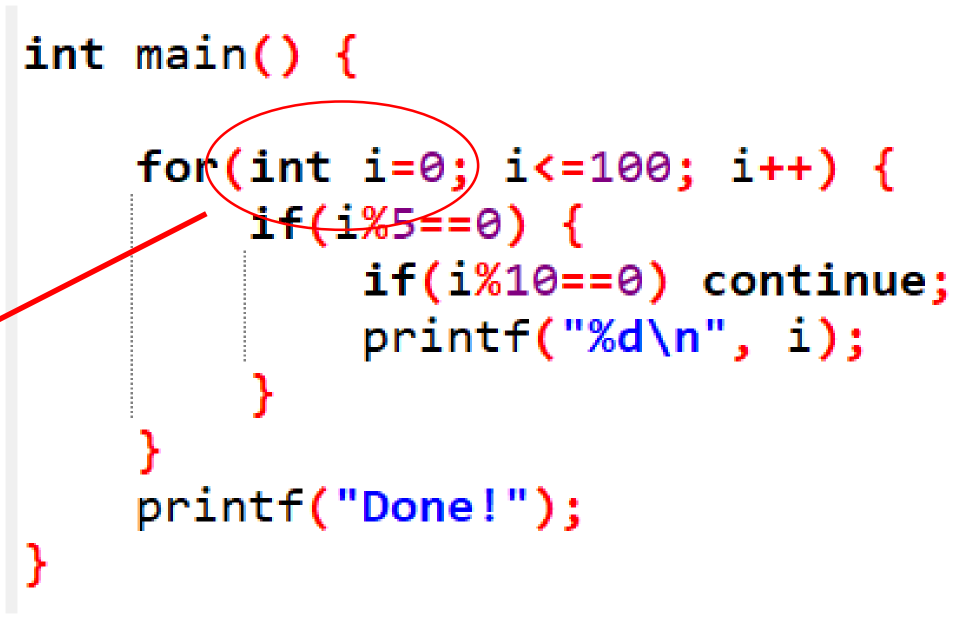


```c
int main() {

    for(int i=0; i<=100; i++) {
        if(i%5==0) {
            if(i%10==0) continue;
            printf("%d\n", i);
        }
    }
    printf("Done!");
}
```

Example: continue

- Write a program to print numbers divided by 5, but not 10 between 1-100.

'for' loop initial declarations are only allowed in C99 or C11 mode

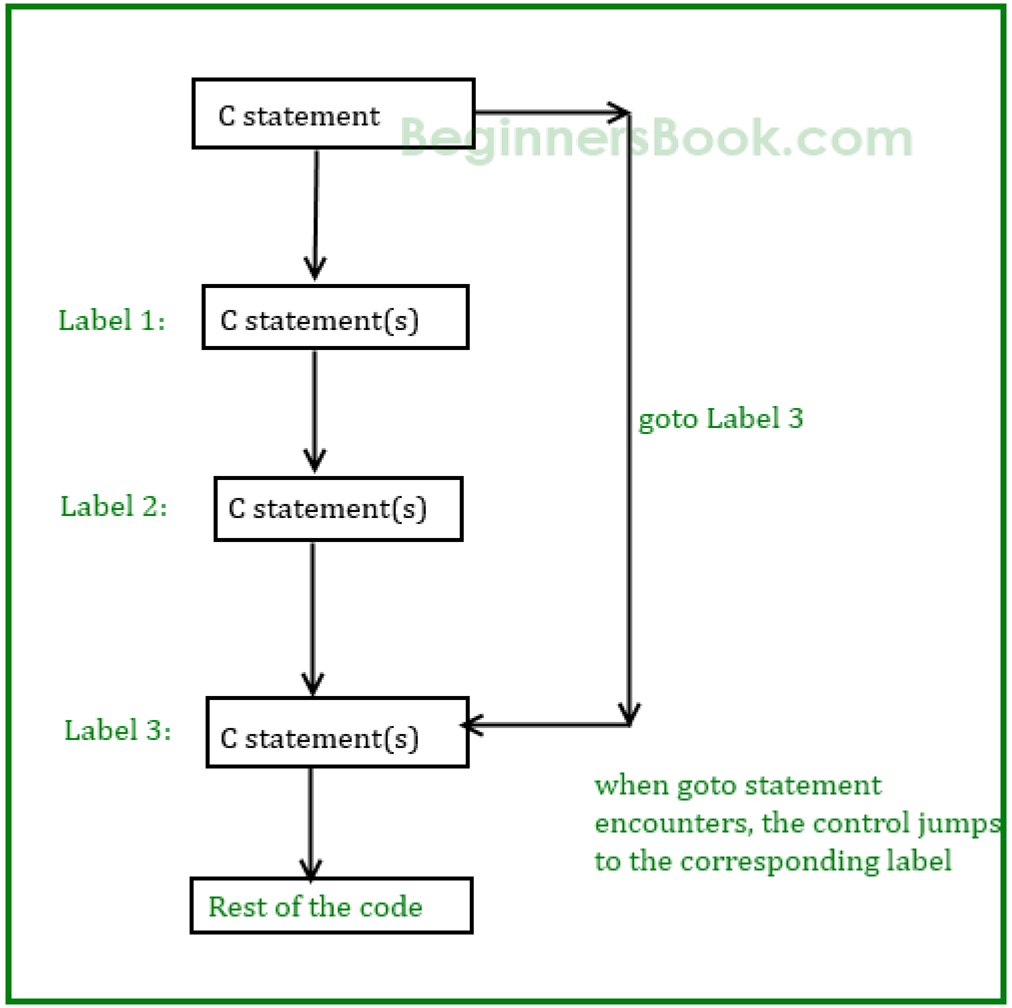


```
int main() {

    for(int i=0; i<=100; i++) {
        if(i%5==0) {
            if(i%10==0) continue;
            printf("%d\n", i);
        }
    }
    printf("Done!");
}
```

the standard C99 is an ISO 9899:1999 version that was a revised C version and denoted as the prime 'C Language'. However, it has was again replaced in 2011 by a higher and more revised standard of C called as C11 (originally C1X) that follows the ISO 9899:2011 standard.

goto

- The goto statement is used to alter the normal sequence of a C program.
- When goto statement is encountered, control of the program jumps to label: and starts executing the code.
- However, The goto statement is rarely used because it makes program confusing, less readable and complex.
    - Also, when this is used, the control of the program won’t be easy to trace, hence it makes testing and debugging difficult.



Example: goto

```
// Program to calculate the sum and average of positive numbers
// If the user enters a negative number, the sum and average are displayed.


int main() {

    const int maxInput = 100;
    int i;
    double number, average, sum = 0.0;

    for (i = 1; i <= maxInput; ++i) {
        printf("%d. Enter a number: ", i);
        scanf("%lf", &number);

        // go to jump if the user enters a negative number
        if (number < 0.0) {
            goto jump;
        }
        sum += number;
    }

jump:
    average = sum / (i - 1);
    printf("Sum = %.2f\n", sum);
    printf("Average = %.2f", average);

    return 0;
}
```

Home work

1. Write a program to take a positive integer (let say n) as an input from the user and calculate the sum of all integers up to n.

2. Write a program to determine the given number is prime number

3. Write a program to return the largest Integer from the given set of numbers.

4. Write a C Program to reverse a number