Noor Saleem

ID # 6811

Subject: Visual Programm

Submitted to

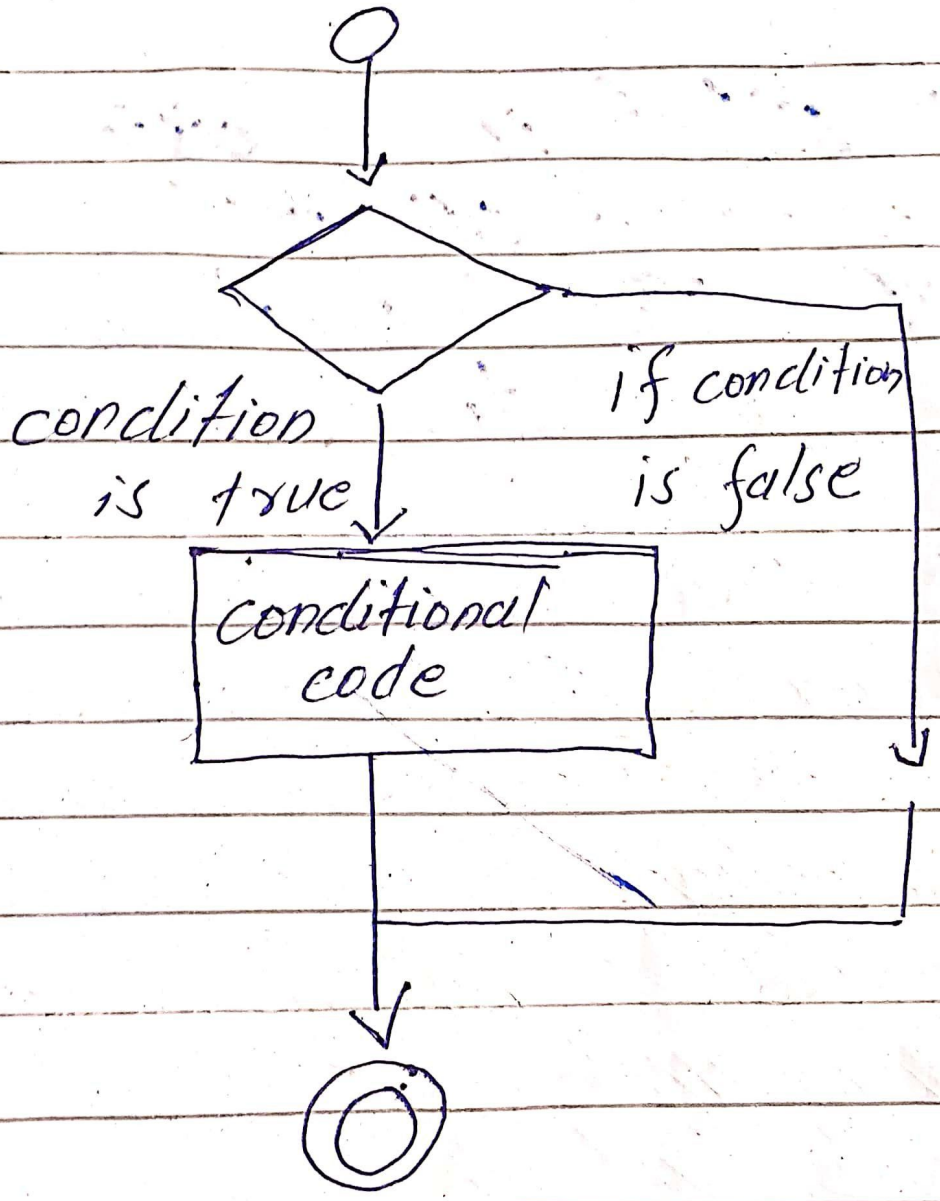
Sir ayub khan.

Q No 1)

(a) What is decision making in C# explain with the help of flowcharts.

Ans
=

Decision making structures requires the programmer to specify one or more conditions to be evaluated or tested by the program. along with a statem or statement to be executed if the condition is determined to be true, and optionally, other statement to be executed if the condition is false.



ON01)

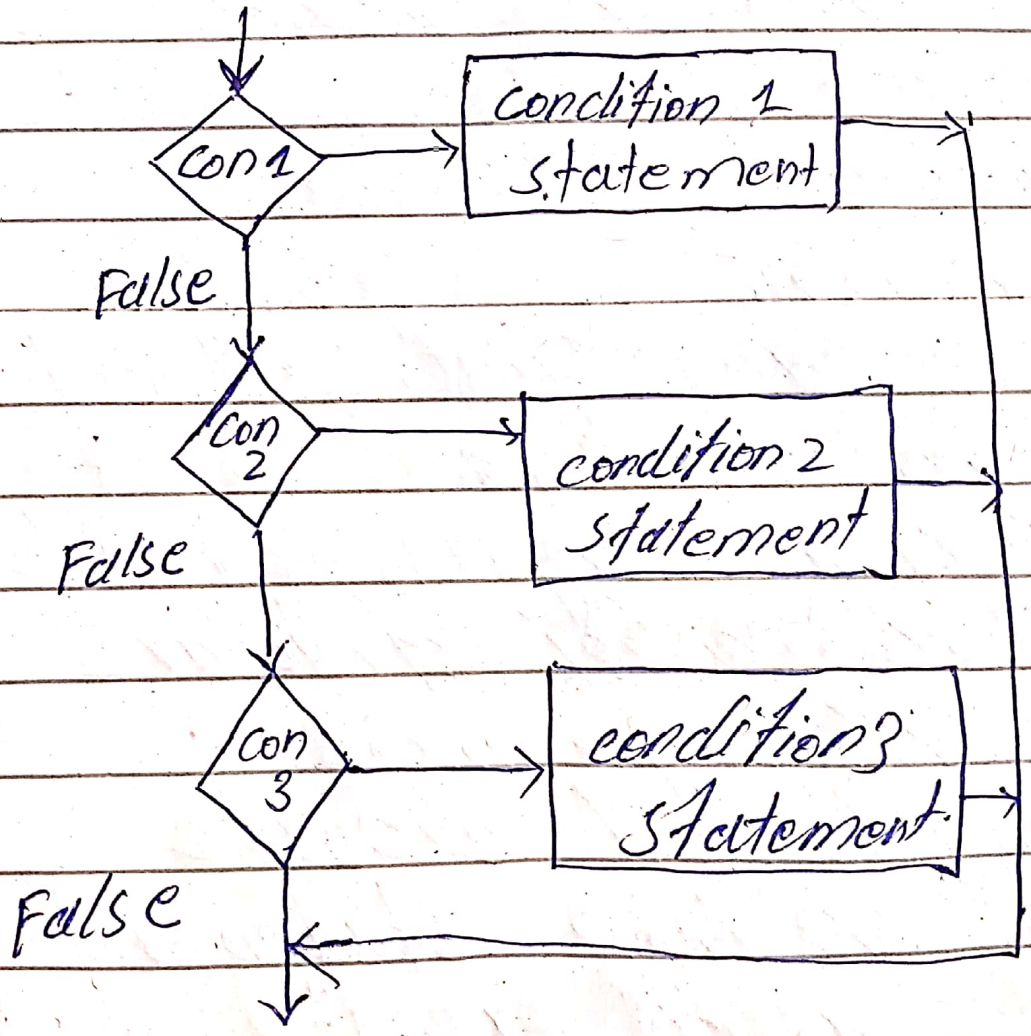
(b) eps

```
#include<stdio.h>
int main()
  char gender;
printf("Enter gander(M/m or
 scanf("%c", &gender);
 switch(gender)
 {
   case 'M':
   case 'M':
 Printf("Female.");
   break;
 default:
printf("Unspecified")
 }
   printf("\n");
   return 0;
}
```

(Q NO 2)

(a) Ans) The if/else statement allows you to create a chain of if statement.

Flow chart:



Q NO2

(b) Ans

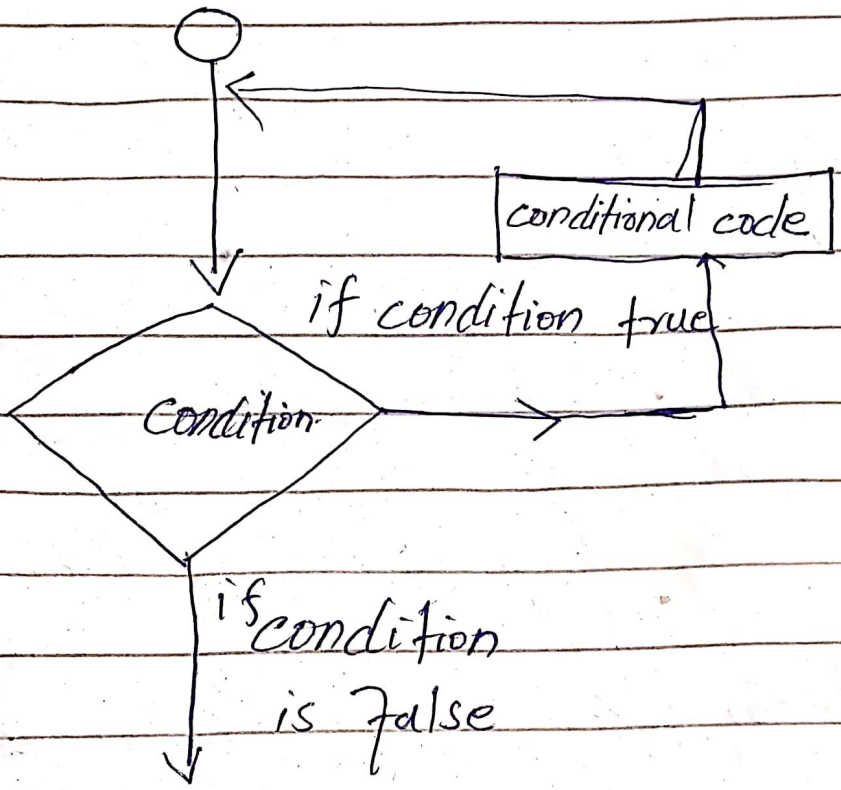
```
#include<stdio.h>
void main()
{
   int temp;
printf("input days temperature:");
scanf("%d", &temp);
   if (temp<0)
   printf("very freezing weather.\n");
   else if (temp<10)
printf("very cold weather.\n");
   else if(temp<20)
   printf("cold weather.\n");
   if (temp<30)
printf("normal in temp.\n");
   else if (temp<40)
printf("its hot.\n");
   else
   printf("its very hot.\n");
```

Q NO3:

(a)

There may be a situation, when you need to execute a block of code several number of times. In general, the statement are executed sequentially: the first statement in a function is executed first, followed by the second, and so on. programing languages provide various control structure that allow for more complicated execute path. A loop statement allow us to execute a statement or a group of statement multiple times and following is the general form

of a loop statement in most of the programming languages.



ON03)

(b) Examples of loops.

(1) while loop.

A while loop statement in C# repeatedly executes a target statement as long as given condition is true.

example:

```
while (condition) {

statement(s);
}
```

(2) For Loop

A for loop is a repetition control structure that allows you to efficiently write a loop that execute a specific number of times.

Syntax:

```
for (init, condition, increament)
{
 statement(s);
}
```

Example:

```
using system;
namspace loops {
class program {
static void Main (string[] args) {
 for (int a=10; a<20; a=a+1) {
console.writeLine("value of a:{0}", a));
```

Q No 4

Ans

When should one used for loop instead of a while loop?
Following loops are identical except for their syntax if so, then why choose one over the other program.

```
int i;
for(i=0; i<arr.length; i++)
{
  // do work
}
int i = 0
while(i<arr.length){
    // do work
    i++;
}
```

Generally use for loop when you have a defined rang of value to iterate over. Use while loop when you are not iterating, or don't know when your exit condition will be. could confuse some people and cause of one error. For loop implicitly a traversal of set, with defined biginng and ending point. In given program when value is end the for loop stop the increament in the given length and program is exit. while loop is continued when your condition is true.

ONO5:

(a) What is encapsulation and its role in Object oriented Programing?

Ans

Encapsulation is defined as The process of enclosing one or more items within a physical or logical package' - Encapsulation, in object oriented programing methodology, prevents access to implementation details. Abstraction and encapsulation are related features in object oriented programing. Abstraction allows making relevant information visible and encapsulation enables a program to implement The desired level of abstraction. Encapsulation is implemented by using

access specifiers. An access specifiers defines the scope and visibility of class member. C# support the following access specifier.

- Public
- Private
- protected
- internel
- protected internel

ONO5)

(b) Why access specifiers are used in encapsulation Justify your answer with help of C# coded example.

Ans (1) Public Access Specifires:

The following example illustrates This.

```
using System;
name space RectangleApplication{
    class Rectangle{
    // member variables
    public double length;
    public double width;

    public double GetArea(){
     return length * width;
    }
    public void Display(){
    console.WriteLine
    ("Length: {0}", length);
    console.WriteLine
    ("width: {0}" width);
    console.writeline
    ("Area: {0}", GetArea());
    }
}//end class Rectangle.
```

```
class ExecuteRectangle{
  static void Main(string[]args){
  Rectangle r = new Rectangle();
  r.length = 4.5;
  r.width = 3.5;
  r.Display();
  Console.ReadLine();
  }
}
}
```

when code compiled and execute

```
Length : 4.5
width : 3.5
Area : 15.75
```

(7) private Access Specifie.

```
using System;
namespace RectangleApplication{
  class Rectangle{
    // member variable
public void Accept details(){
```