**Name: Afaq Ahmad Shah**

**ID # 15715**

**Paper: Programming Fundamental**

**Teacher: Dr. Fazal-e-Malik**

**Question:1**

**a): What is the purpose of if statement? Discuss its two different forms with examples.**

**Answer:**

**Purpose of if statement:**

The if statement is used to check a condition and if the condition is true, we run a block of statements (called the if-block), else we process another block of statements (called the else-block). The else clause is optional.

**Example of if statement:**

Explanation: The condition (x<y) specified in the “if” returns true for the value of x and y, so the statement inside the body of if is executed.

if statement

**if statement**

(this condition is true )

execute this statement ;

•Single-entry/single-exit

•Nonzero is true, zero is false

**C – If statement**

Syntax of if statement:

The statements inside the body of “if” only execute if the given condition returns true. If the condition returns false then the statements inside “if” are skipped

The if-else Statement

The if-else

Different actions if conditions true or false

•Example

if student’s grade is greater than or equal to 60

Display message “Passed”

else

Display message “Failed”

•C++ Code

if ( grade >= 60 )

cout<<("Passed“);

else

cout<<("Failed“);

flow diagram of if statement:



**Flow chart:**

****

**Example of if statement:**

****

****

**Example of multiple if statement:**

****

****

**b)**

**Write a C++ program to read two numbers from keyboard and then find the**

**LARGEST number of them.**

**Answer:**

****

**Question:2**

1. **What are the logical operators? Explain them.**

**Answer:**

**Logical Operators**

•&& (logical AND)

 –Used to combine two conditions

 –true if both conditions are true

 if ( gender == 1 && age >= 65 )

 senior++;

•|| (logical OR)

 –true if either of condition is true

 if (semesterAvg >= 90 || finalExam >=90 )

 cout<<("Student grade is A”);

**Cont.**

•! (logical NOT, logical negation)

 –Returns true when its condition is false, & vice versa

 if ( !( grade == 20 ) )

 cout<<“hello world“);

Alternative:

 if ( grade != 20 )

 cout<<“hello world“);

**Logical Operators..**

|  |  |
| --- | --- |
| Operands | Results |
| X0 | Y0 | !x1 | !y1 | X&&y0 | X||y0 |
| **0** | **Non-zero** | **1** | **0** | **0** | **1** |
| **Non-zero** | **0** | **0** | **1** | **0** | **1** |
| **Non-zero** | **Non-zero** | **0** | **0** | **1** | **1** |

**Sample Program**

•To calculate the division

•Input: marks of 2 different subjects

•Rules

–Percentage above or equal to 60 - First division

–Percentage between 50 and 59 - Second division

–Percentage between 40 and 49 - Third division

–Percentage less than 40 – Fail

•Solution

–Nested if-else

Go to Program

–Logical operators

Go to Program

**Using Logical Operators**

•A company insures its drivers in the following cases:

–If the driver is married

–If the driver is unmarried, male & above 30 years of age

–If the driver is unmarried, female & above 25 years of age

 **Using Logical Operators..**

if ( ms == 'M' )

 cout<<"Driver is insured";

else

 {

 if ( sex == 'M' )

 {

 if ( age > 30 )

 cout<<"Driver is insured" ;

 else

 cout<<"Driver is not insured" ;

 }

 else

 {

 if ( age > 25 )

 cout<<"Driver is insured" ) ;

 else

 cout<<"Driver is not insured" ) ;

 }

}

if ( ( ms == 'M') || ( ms == 'U' && sex == 'M' && age > 30 ) ||

( ms == 'U' && sex == 'F' && age > 25 ) )

 cout<<"Driver is insured" ) ;

else

 count<<“Driver is not insured”;

**Salary Calculation Example:**

|  |  |  |  |
| --- | --- | --- | --- |
| Gender | Years of service | qualifications | Salary |
| male | >=10 | Post-graduate | 15000 |
|  | >=10 | Graduate | 10000 |
|  | <10 | Post-graduate | 10000 |
|  | <10 | graduate | 7000 |
| female | >=10 | Post-graduate | 12000 |
|  | >=10 | Graduate | 9000 |
|  | <10 | Post-graduate | 10000 |
|  | <10 | Graduate | 6000 |

**Salary Calculation Example..**

char g ;

int yos, qual, sal ;

cout<<"Enter Gender, Years of Service and Qualifications ( 0 = G, 1 = PG ):";

cin>> g,yos,qual ;

if ( g == 'm' && yos >= 10 && qual == 1 )

 sal = 15000 ;

else if ( ( g == 'm' && yos >= 10 && qual == 0 ) ||( g == 'm' && yos < 10 && qual == 1 ) )

 sal = 10000 ;

else if ( g == 'm' && yos < 10 && qual == 0 )

 sal = 7000 ;

else if ( g == 'f' && yos >= 10 && qual == 1 )

 sal = 12000 ;

else if ( g == 'f' && yos >= 10 && qual == 0 )

 sal = 9000 ;

else if ( g == 'f' && yos < 10 && qual == 1 )

 sal = 10000 ;

else if ( g == 'f' && yos < 10 && qual == 0 )

 sal = 6000 ;

**Revised Hierarchy:**

|  |  |
| --- | --- |
| Operators | Types |
| !/ % + -< > <= >=== !=&&||= | Logical NOTArithmetic and modulusArithmeticRelationalRelationalLogical ANDLogical ORAssignment |

**Confusion**

•Between assignment /equality operator

if ( i = 5 )

 cout<<"You entered 5";

else

 cout<<"You entered something other than 5";

•Null statement

 if ( i == 5 ) ;

cout<<"You entered 5"

**b)**

**Write a C++ program to get Temperature in Fahrenheit F and then find the Atmosphere according to the below rules:**

**• If temperature F is above 40 degree Fahrenheit then display…………………..Very Hot.**

**• If temperature F is between 35 & 40 degree Fahrenheit then display………Tolerable.**

**• If temperature F is between 30 & 35 degree Fahrenheit then display………Warm.**

**• If temperature F is less than 30 degree Fahrenheit then display………….……Cool.**

**Amswer:**

****

**Q.3**

**a)**

**What does Looping mean? Explain different loops in C++.**

**Answer:**

**Loops**

A shape produced by a curve that bends round and crosses itself.

Example program 1

•Three numbers x, y, z are input through the keyboard, write a program to determine the smallest of the three.

Write a program

false

true

 X<Y

true

Z<y

false

X<y

true

true

Display y is smallest

Display z is smallest

Display x is smallest

**Example program 2**

•Any character is entered through the keyboard, write a program to determine whether the character entered is a capital letter, a small case letter, a digit or a special symbol.

Write a program

|  |  |
| --- | --- |
| Characters  | ASCII Values |
| A-Za-z0-9Special symbols | 65-9097-12248-570-47,58-64,91-96,123-127 |

**Loops in C**

•Loop structure is used to execute a statement or statements repeatedly

•Three major loops structures in C

–while loop

–for loop

–do while

**Loop in general**

•Loop has a termination point  finite loop

–Loop execution stops when the loop condition becomes false

•Loop has a counter that counts number of iterations of that loop

•Loop has a statement or a set of statements that are executed until the loop condition become false

**Repetition-Example:**

•Formula 1 car race

•There is a path/track

•Each car has to complete a certain no of rounds say 10

•In each round, when a car cross the finish line

–the condition is check whether the car has completed total no of round or not.

**General form while loop:**

Initialize loop counter;

While (test loop counter using a condition)

{

 do this;

 and this;

 increment loop counter;

}

**The while loop:**

 START

Count=1

Count=count+1

 PRINT

 si

Si=p\*n\*r\*/100

Is

Count<=3

stop

INPUT

P,n,r

**A while loop program:**

main( )

 Memory

 Count

 Total

{

 int count = 0;

 int total = 0;

 while(count < 5)

 {

 total = total + count;

 cout>>“count =, total = ”, count, total;

 count = count +1;

 }

Program Output

 count = 0, total = 0

count = 1, total = 1

count = 2, total = 3

count = 3, total = 6

count = 4, total = 10

**Points to remember**

•Loop body will keep on executing until the loop condition become false

•When loop condition become false, the first statement after the while block will be executed

•Condition can be a single or compound

 While (i<=10)

 While (i>=10&&j<=15)

 While (j>10&&(b<15||c<20))

**Cont.**

•Statement within loop body can be single line or block of statement.

•In case of single line parentheses are optional

 While (i<=10)

 i=i+1;

 is same as while (i<=10)

 {

 i=i+1;

 }

•In case of block, parentheses are must

**Cont..**

•A loop can be infinite loop

Main() main ()

{ {

 Int i=1; int i=1;

 While (i<=10) while (i<=10)

 {

Count<<“\n”,i;

Count<<“\n”,i;

 } i=i+1;

 }

 }

**Cont.**

•Loop counter can be decremented

Main()

{

 Int i=5

 While (i>=1)

 {

 Count<<“\nMake the computer literate”;

 i=i-1;

 }

 }

**Cont.**

•It is not necessary that a loop counter must only be int. it can be float or char

 Main()

 {

 Float a=10.0;

 While (a <=10.5)

 {

 cout<<“\nRaindrops on roses…”;

 cout<<“…and whiskers on kittens”;

 a=a+1;

 }

 }

**Pre and post increment operator**

•Pre increment/ decrement

–++x; is same as x = x + 1;

–or – –x is same as x = x – 1;

•Post increment/ decrement

–x++; is same as x = x + 1;

–or x-- is same as x = x – 1;

**Difference between pre and post operators:**

x=1;

y=x++

y=x;

x=x+1;

 Count<<“x=,y=”,x,y);

Output

X=2, y=1

x=x+1;

y=x;

x=1;

y=++ x;

 Count <<“x= ,y=” , x,y);

Write a program

Output

x=2,y=2

**Other operator:**

 Main()

 {

 Int i=1;

 While (i<=10)

 {

 Count<<“\n”,i;

i+=1;

 }

i=i+1;

 }

•+= is a compound assignment operator

•j = j + 10 can also be written as j += 10

•Other compound assignment operators are -=, \*=, / =, %=.

Example program:

Main()

{

 Int i=0;

First comparison i<10 is performed then value of i is incremented

 While (i++<10)

 Count<<“\n”,i;

 }

Main()

{

First the value of i is incremented then comparison i<10 is performed.

 Int i=0;

 While (++i<=10)

 Count<<“\n”,i;

}

**for loop**

•The most popular looping instruction

•The for loop allows us to specify three things in a single line

1.Initialization:

•Setting a loop counter to an initial value

2.Condition:

•Testing loop counter to check whether it reaches the desired number of repetitions

3.Increment / decrement part:

•Increase or decrease the value of loop counter each time when the statement within the loop are executed

**General form of for loop:**

 for ( initialization condition incr\decr)

 {

 do this;

 and this;

 and this;

 }

**Cont.**

•It is important to note that the initialization, testing and incrementation part of a for loop can be replaced by any valid expression.

For (i=10; i; i--)

 Count<<i;

For (i<4;j=5;j=0)

 Count<<i;

**Ways for writing for loop:**

**a)**Main ()

 {

 Int i;

 For (i=1;i<10;i=i+1)

 Count<<“\n”,i,;

 }

 b)main()

 {

 Int i;

 For (i=1;i<=10;)

 {

 Count<<“\n”,i;

 i=i+1;

 }

 }

Cont

e)main()

{

 Int i;

 For (I=0;i++<10;)

 Count<<“\n”,i;

 }

f)main()

 {

 Int i;

 For (i=0; ++i<=10;)

 Count<<“\n”,i;

 }

**Nested for loop:**

/\* Demonstration of nested loops \*/

main( )

{

 int r, c, sum ;

 for ( r = 1 ; r <= 3 ; r++ ) /\* outer loop \*/

 {

 for ( c = 1 ; c <= 2 ; c++ ) /\* inner loop \*/

 {

 sum = r + c ;

 printf ( "r = %d c = %d sum = %d\n", r, c, sum ) ;

 }

 }

}

Example program 1:

•Write a program to find the factorial value of any number entered through the keyboard.

start

Display input a number

stop

Diplay f

x--

f=f\*x

x>1

Read x

F=1



**Example program 2**

•Write a program that displays the table of x up to y. The value of x and y is input by the user.

 e.g. x = 4 and y = 5

4 x 1 = 4

 4 x 2 = 8

 4 x 3 = 12

 4 x 4 = 16

 4 x 5 = 20



**The break Statement**

•There are situations where we want to jump out of a loop instantly, without waiting to get back to the conditional test.

•The keyword break allows us to do this

•When break is encountered inside any loop, control automatically passes to the first statement after the loop.

•A break is usually associated with an if.

•The keyword break, breaks the control only from the loop in which it is placed

**Cont.**

•Example: Write a program to determine whether a number is prime or not. A prime number is one, which is divisible only by 1 or itself.



break statement in nested loop

main()

{

 int i = 1 , j = 1 ;

 while ( i++ <= 100 )

 {

 while ( j++ <= 200 )

 {

 if ( j == 150 )

 break ;

 else

 count<<“\n”,I,ju;

 printf ( "%d %d\n", i, j ) ;

 }

 }

}

**The continue Statement**

•continue statement allows to take the control to the beginning of the loop, bypassing the statements inside the loop, which have not yet been executed

•A continue is usually associated with an if.



The do-while Loop

•Post test repetition structure

•The statement in the loop block are executed at least once, whether the loops condition is true or false

do

{

 this ;

 and this ;

 and this ;

} while ( this condition is true ) ;

**General form of do while loop:**

****

Main()

{

 While (4<1)

 Cout<<“hello there\n”;

}

Main()

{

 do

 {

 Cout<<“hello there\n”;

 } while (4<1);

}

**Example program:**

****j=j+i;

Program output

2

5

i=i+j;

 **b)**

**Write a C++ program to read a number from keyboard and then determine whether it is Even or Odd number?**

**Answer:**

output odd number:



Output even number:



**Q#4 :What is the purpose of using break and continue statements?**

**ANSwer:**

C break and continue

We learned about loops in previous tutorials. In this tutorial, we will learn to use break and continue statements with the help of examples

**C++ Break**

You have already seen the break statement used in an earlier chapter of this tutorial. It was used to "jump out" of a switch statement.

The break statement can also be used to jump out of a loop.





C++ Continue

The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.



Out put:



Answer4 (B)

****

****

**Q.5**

**Expain the the following with proper examples**

**a) C++ Character set**

**b) Constants**

**c) Variables**

**d) Keywords**

**e) Relational Operators**

**answer:**

****










Constants

Integer Constants

Rules:

•Also called Floating Point constants

•A real constant must have at least one digit

•It must have a decimal point

•It could be either positive or negative

•Default sign is positive

•No commas or blanks are allowed within a real

constant

•Examples: +325.34, 426.0, -32.76, -48.5792

Character Constants

•A character constant is a

-single alphabet

-a single digit

-or a single special symbol

•Enclosed within single inverted commas, ‘ ’

•Both the inverted commas should point to the

 left

•The maximum length can be 1 character

 Examples: ’A’, ‘I’, ‘5’, ‘=‘

C) Variables

Variables..

Variables..

•No commas or blanks are allowed within a

variable name.

•No Special symbol are used in name.

•Examples: Interger1, Sum, \_FirstNum

•Invalid variable names

-#sum, 12x, first name

D) Keywords



e)relation operation:



 The end…..