Name Farkhanda Jabeen

Id 14725

Paper Programming fundamental

Teacher Dr. Fazal-e-Malik

Date 29/sep/2020

Farkhanda Id : 14725

Q No 1 (a)

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.

There are two types of if Condition they are IF - THEN and If - THEN - EISE

// IF Condition.

```
#include <iostream>
main()
{
    if (condition)
    {
        // Body of if Statement
    }

}
```

Farkhanda Id:14725

// If-THEN -ELSE Condition

```
main()
{
    if (condition)
    {
        // Block of code if Condition is true
    }
    else {
        // Block of code if condition is false
    }
}
```

OR

If (condition)

do this

IF-THEN-ELSE

If (condition)

do this

else

do this

Farkhanda Id : 14725

Q NO 1 (b)

```
#include <iostream.h>
#include <conio.h>
Using namespace std;

int main ()
{
   int n1, n2;

   Cout<< "Enter two numbers from keyboard : \n" ;
   Cout << "enter first number \n";
    Cin >> n1;
    Cout << "enter 2nd number \n";
     Cin >>n2;

 if (n1 >=n2)
    {
      Cout << "largest number is:";
      Cout << n1;
    }
     else {
       Cout << "Largest number is :";
       Cout << n2;
     }

      return 0;
}.
```

Farkhanda Jabeen Id 14725

Qno 2 part (A)

Q What are logical operators? Explain them.

Ans Logical operators:-
Logical Operators are used with binary variables. They are mainly used in Conditional Statements.

Logical operators in C++ are &&, ||, !.

* && (logical AND)
It is used to combine two Condition.
→ True if both Conditions are true

```
if (gender == 1 && age >= 74)
```

* || (Logical OR)
• True if either of a Condition is true.

```
(a == b) || (c < b)
```

in this example only one condition is true so its answer will also be true.

* ! (logical NOT)
• Returns ~~true~~ true when its condition is false, & vice versa.
→ True only if operand is 0

```
if C = 5 then expression
! (C == 5)
equals to zero.
```

Farkhand jabeen Id : 14725

Q No 2 part (b)

```
#include <iostream>
#include <conio.h>
using namespace std;

  main()
{
      int tmp;

    cout <<" temperature is \n";
    cin >> tmp;
    if (tmp >= 40)
    {
      cout <<" its very hot \n";
    }
    else if (temp > 35 && temp < 40)
    {
      cout <<" its tolerable \n";
    }
     else if (tmp >= 30 && temp <= 35)
     {
       cout <<" its warm \n";
     }
     else
     {
        cout << "Cool";
     }
}
```

Farkhanda
Id 14725 (6)

QNO3 (a)

Looping:-

* In Computer Science, a loop is a programming structure that repeats a sequence of instructions until a specific condition is met.
* Loop structure is used to execute a statements repeatedly.

Types of Loops:

There are three types of loops

* while loop
* for loop
* do while

While loop:-

a while loop is a control flow statement that allows code to be executed repeatedly based on Boolean condition.

General form of while loop:-

```
initialise loop counter;
while ( test loop counter using a condition.
{
    do this;
    and this;
    increment loop counter;
}
```

Farkhanda Id: 14725

For loop:-

For loop is the most popular looping instruction.

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

1) Initialization
2) Condition
3) Increment /decrement

General form of for loop

```
For (initialization; condition; incr/decr)
{
    do this;
    and this;
    and this;
}
```

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);
```

(8)

Farkhanda Id 14725

Q No 3 part (b)

```
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
    int number;

    cout << "Enter the number\n";
    cin >> number;
    if (number % 2 == 0)
        cout << number << " is an even number";
    else
        cout << number << " is an odd number";

    return 0;

}.
```

Q No 4 part (a)

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

* **The Continue Statement:-**

* Continue statement allows to take the control to the beggining of the loop, by passing the statements inside loop, which have not yet been executed.

* A continue is usually associated with an if.

(10)

Farkhand a id 14725

Q NO 4 Part (b)

```
#include <iostream>
#include <conio.h>
Using namespace std;

   int main()
{
      int i;
      int sum = 0;

   Cout<<"The first 10 natural numbers are: \n";
    for (i = 1; i<=10; i++)
    {
      Cout <<i<< "   ";
       Sum = sum +i;
    }
      Cout <<"\n The Sum of first 10 natural numbers is \n";
      cout <<sum;
       return 0;

}.
```

Q NO 5

Explain the following with proper examples.

a) **C++ Character Set**

Ans In c++ character set there are

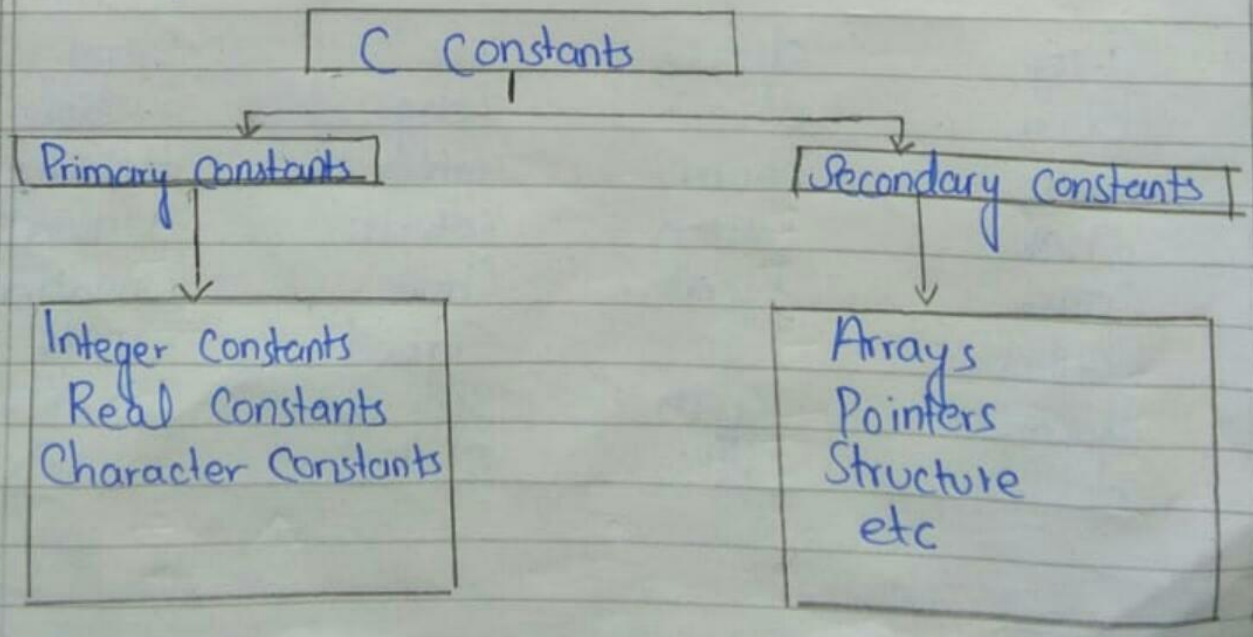
Alphabets :- A, B .... Y, Z.
a, b, c ..... y, z.

Digits :- 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Special Symbols :-
~, !, #, { }, [ ], : , ; , " , ' ,
?, . , / , | , \ , $, % , ^ , & , * , ( ) , _ ,
= , + .

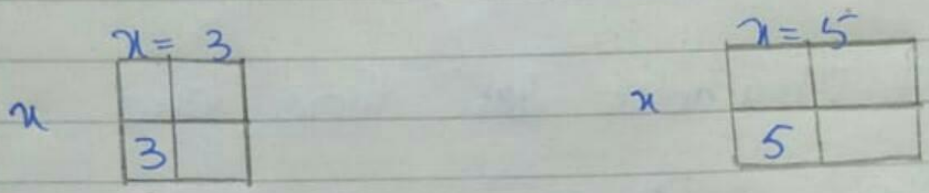
b) **Constants :-**

- Constant is an entity that doesn't change.



c) Variables :-

* An entity that may change during program execution.
* These are names given to locations in memory.



* Series of characters (letters, digits, underscores).
* Must begin with a letter or underscore.
* Case Sensitive.
* Meaningful naming Scheme.

Examples. : Integer1, Sum, FirstNum.

d) Keywords :-

* These are reserved words.
* Compiler knows their meaning
* Cannot be used as variable name
* Cannot be changed

auto	double	int	Struct
break	else	long	Switch
Case	enum	register	typedef
Char	extern	return	Union
Const	float	Short	Unsigned
Continue	for	Signed	void
default	goto	Size of	volatile
do	if	Static	while

Farkhanda id 14725.

(13)

Relational operators

Standard Algebric Relational operator	C++ equality	Example	Meaning
$>$	>	x > y	x is greater than y
$<$	<	x < y	x is less than y
$\geq$	>=	x >= y	x is greater than or equal to y.
$\leq$	<=	x <= y	x is less than or equal to y
Equality operators			
=	==	x == y	x is equal to y
$\neq$	!=	x != y	x is not equal to y