Name: Aftab Khan ID:12985
Subject: Programming Fundamental

Answer 1(a):

if Statement Purpose:

C++ has the following **conditional statements**: **Use if** to specify a block of code to be executed, **if** a specified condition is true.

Use else to specify a block of code to be executed, **if** the same condition is false. **Use else if** to specify a new condition to test, **if** the first condition is false.

Syntax:

```
if(condition){

      // set of instruction when condition is true

}
```

We use if statement in two forms and are as follows:

- if-else
- if-elseif-else

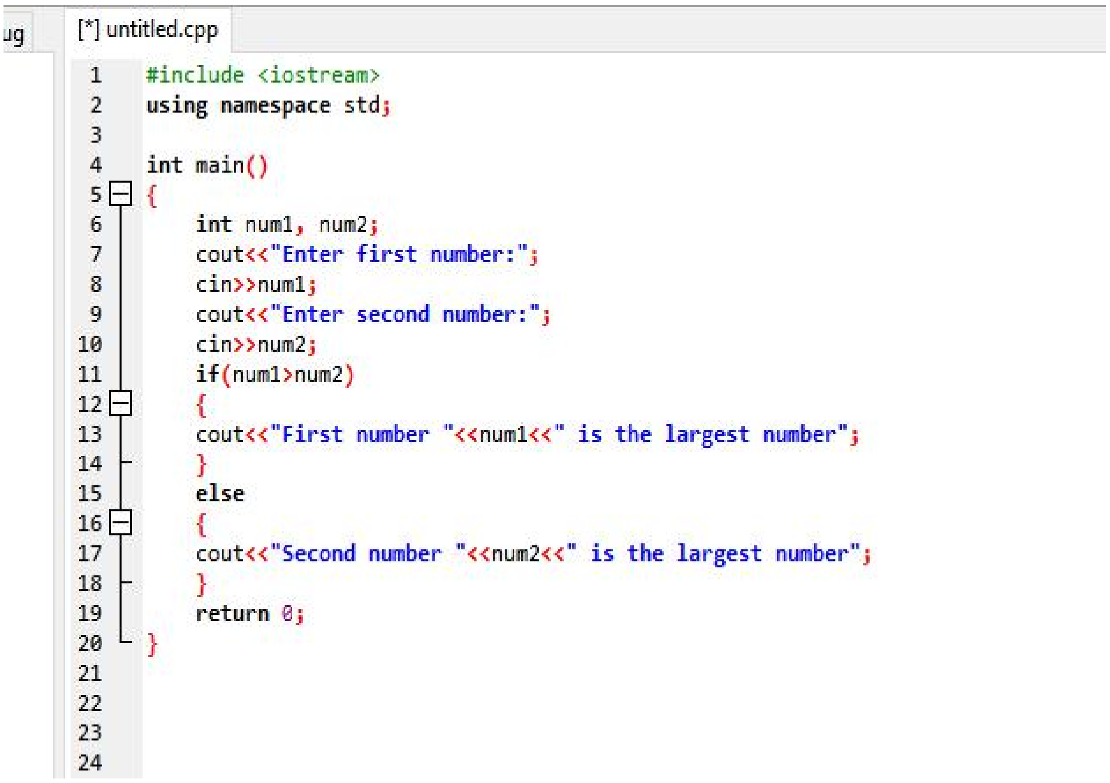
if-else block:

```
if(number>0){

        printf("Number is positive");

}

else{

      printf("Number is not positive");

}
```

if-elseif-else block:

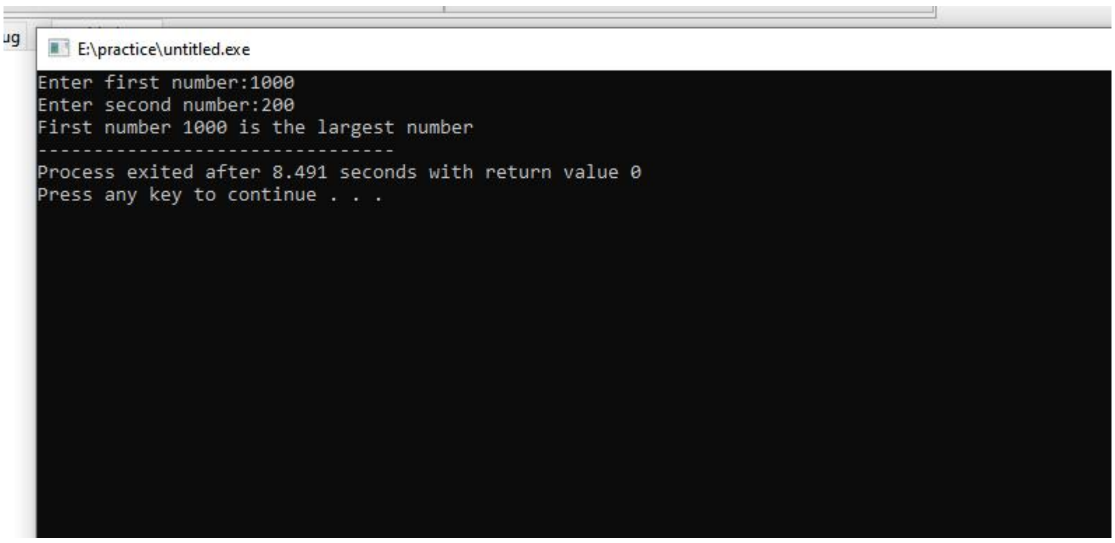
```
if(number%2==0){
        printf("Number is a multiple of 2");
}
else if(number%3==0){
        printf("Number is a multiple of 3");
}
else{
        printf("Number is neither a multiple of 2 nor 3");
}
```

Answer 1(b):

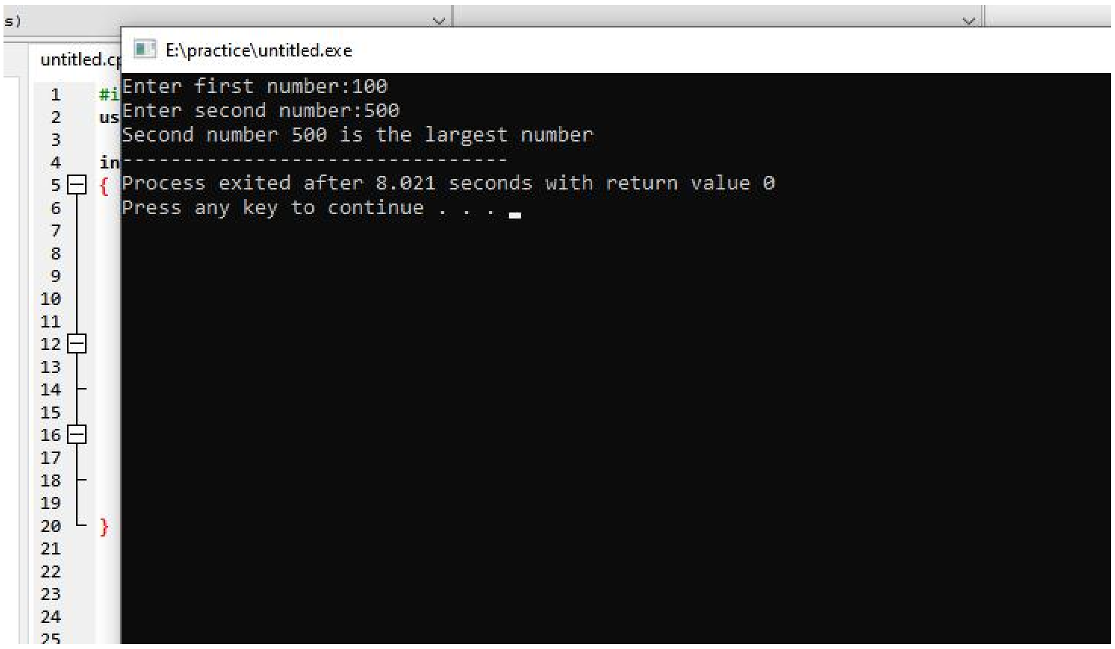


ug   [*] untitled.cpp

```
#include <iostream>
using namespace std;

int main()
{
    int num1, num2;
    cout<<"Enter first number:";
    cin>>num1;
    cout<<"Enter second number:";
    cin>>num2;
    if(num1>num2)
    {
    cout<<"First number "<<num1<<" is the largest number";
    }
    else
    {
    cout<<"Second number "<<num2<<" is the largest number";
    }
    return 0;
}
```



```
Enter first number:1000
Enter second number:200
First number 1000 is the largest number
--------------------------------
Process exited after 8.491 seconds with return value 0
Press any key to continue . . .
```



```
Enter first number:100
Enter second number:500
Second number 500 is the largest number
--------------------------------
Process exited after 8.021 seconds with return value 0
Press any key to continue . . .
```

Answer 2(a):

Logical Operators:

Logical operators play a significant role in any programming language and they are important as they help us to take decisions based on certain conditions.

Types of Logical Operators:

There are three types of Logical Operators:

- Logical AND
- Logical OR
- Logical NOT

Explanation:

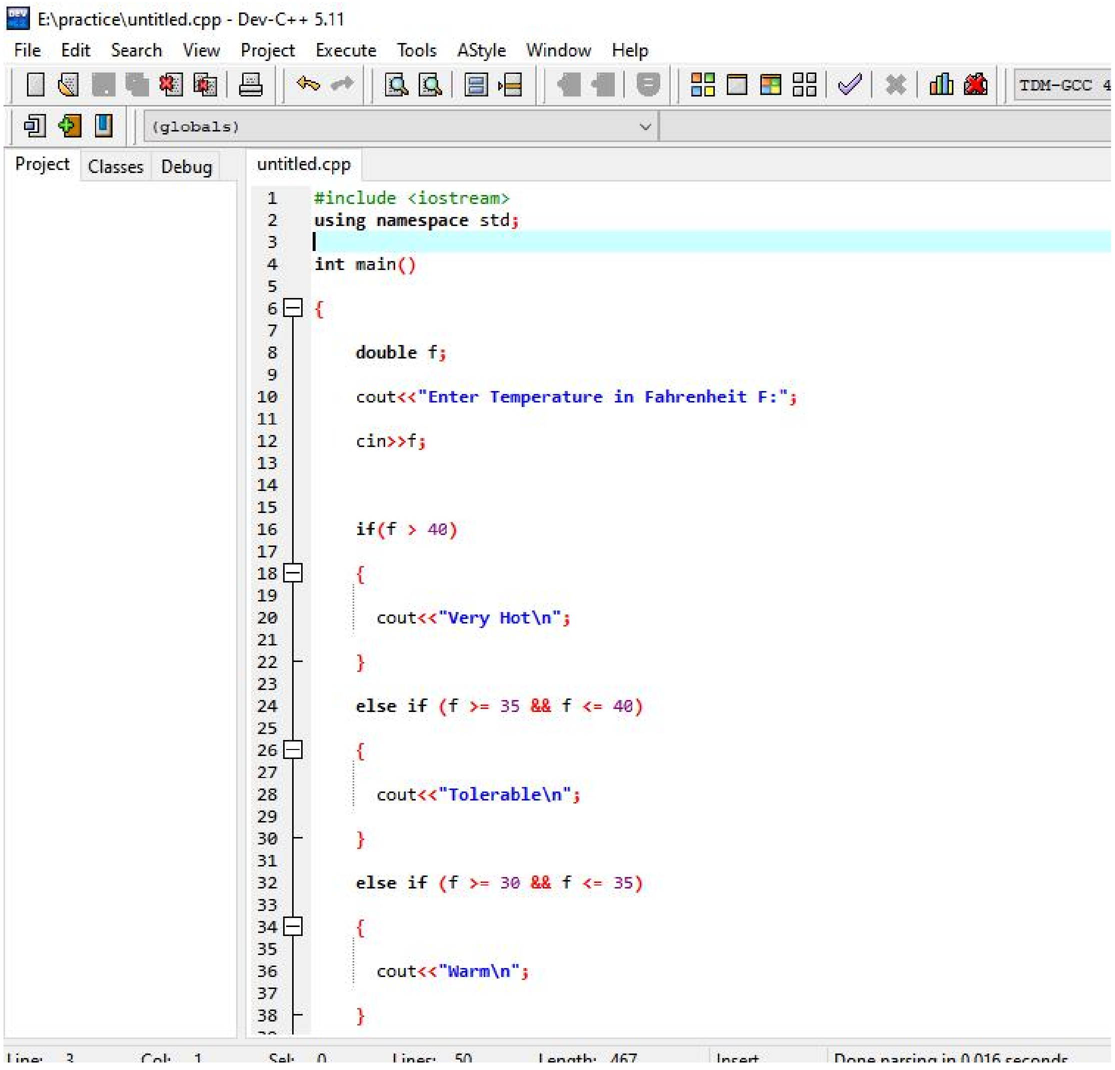
Logical AND (&&): If both the conditions are true then, it will execute the statements.

Logical OR (||): If any one of the conditions is true, then it will execute the statements.

Logical NOT (!): If the condition is true, this operator will make it false.

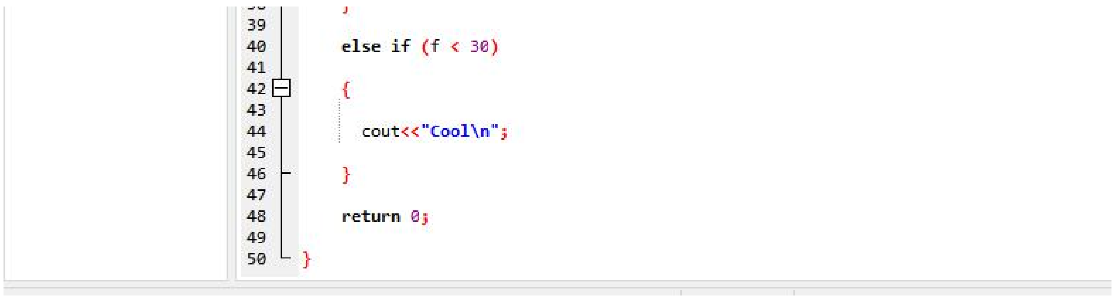
Answer 2(b):

CODE:


E:\practice\untitled.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
TDM-GCC 4
Project Classes Debug
untitled.cpp

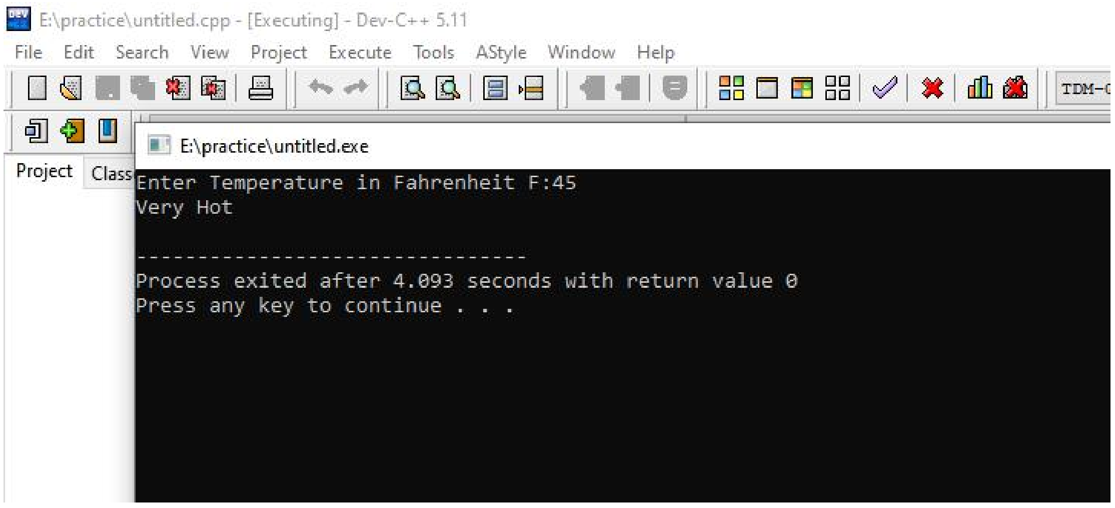
```
#include <iostream>
using namespace std;

int main()

{

    double f;

    cout<<"Enter Temperature in Fahrenheit F:";

    cin>>f;



    if(f > 40)

    {

      cout<<"Very Hot\n";

    }

    else if (f >= 35 && f <= 40)

    {

      cout<<"Tolerable\n";

    }

    else if (f >= 30 && f <= 35)

    {

      cout<<"Warm\n";

    }
```

Line: 3 Col: 1 Sel: 0 Lines: 50 Length: 467 Insert Done parsing in 0.016 seconds



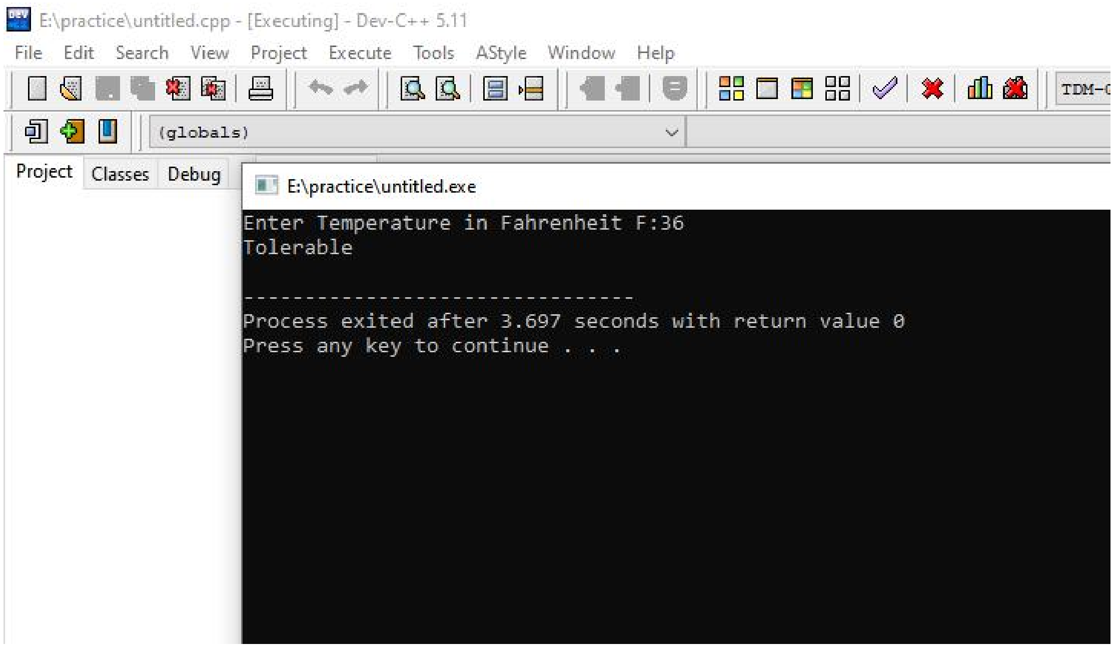
```

    else if (f < 30)

    {

      cout<<"Cool\n";

    }

    return 0;

}
```

Hot:


E:\practice\untitled.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Project Class
E:\practice\untitled.exe

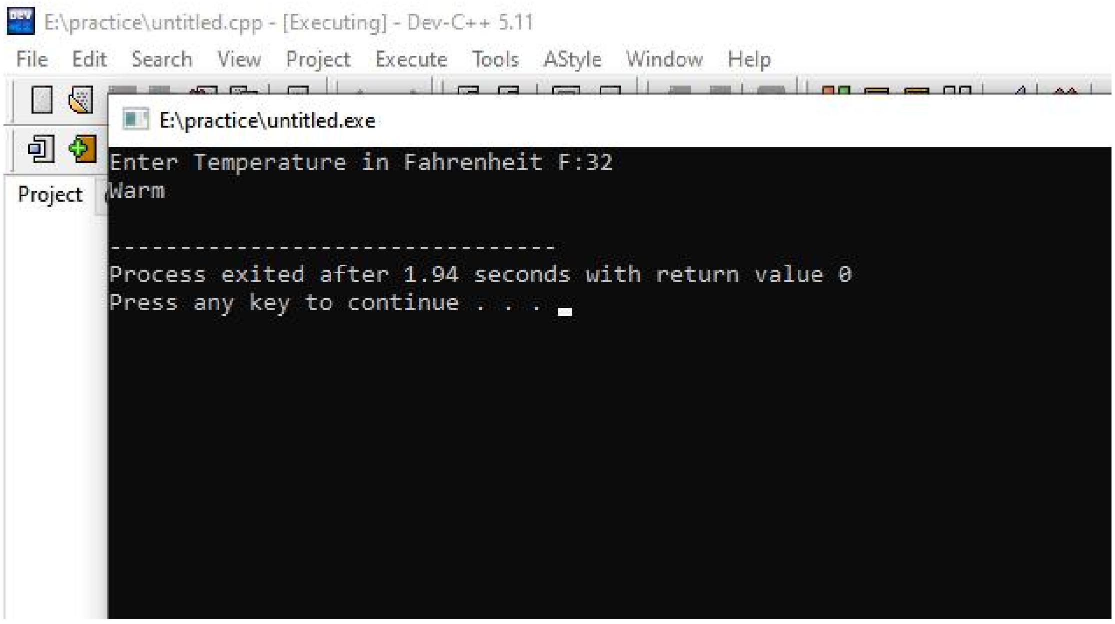
```
Enter Temperature in Fahrenheit F:45
Very Hot

--------------------------------
Process exited after 4.093 seconds with return value 0
Press any key to continue . . .
```

Tolerable:


E:\practice\untitled.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug
E:\practice\untitled.exe

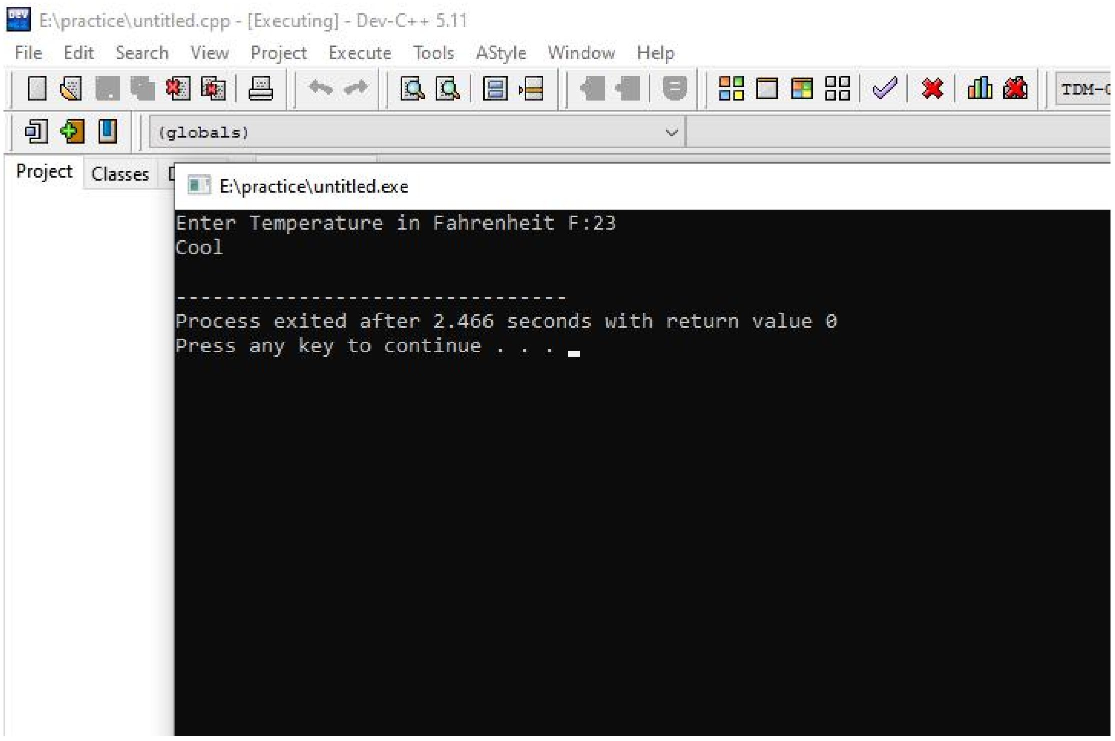
```
Enter Temperature in Fahrenheit F:36
Tolerable

--------------------------------
Process exited after 3.697 seconds with return value 0
Press any key to continue . . .
```

Warm:


E:\practice\untitled.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Project
E:\practice\untitled.exe

```
Enter Temperature in Fahrenheit F:32
Warm

--------------------------------
Process exited after 1.94 seconds with return value 0
Press any key to continue . . .
```

Cool:


E:\practice\untitled.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes
E:\practice\untitled.exe

```
Enter Temperature in Fahrenheit F:23
Cool

--------------------------------
Process exited after 2.466 seconds with return value 0
Press any key to continue . . .
```

Answer 3(a):

Looping:

In computer science, a loop is a programming structure that repeats a sequence of instructions until a specific condition is met. Programmers use loops to cycle through values, add sums of numbers, repeat functions, and many other things.

Loops are supported by all modern programming languages, though their implementations and syntax may differ.

In C++ language there are three types of loops are used.
They are:

1) For loop
2) While loop
3) Do-while loop

1) For loop:

The syntax of for loop is

```
for ( initial statement; condition; increment or decrement)
{
    //body of the for loop
    //statement1
    //statement2
}
```

2) While loop:

In this loop, the condition is to check first then the body of the loop is executed.
It is entry-controlled loop.

The syntax of for loop is

```
while(condition)
{
    //body of the while loop
    //statement1
    //statement2
}
```

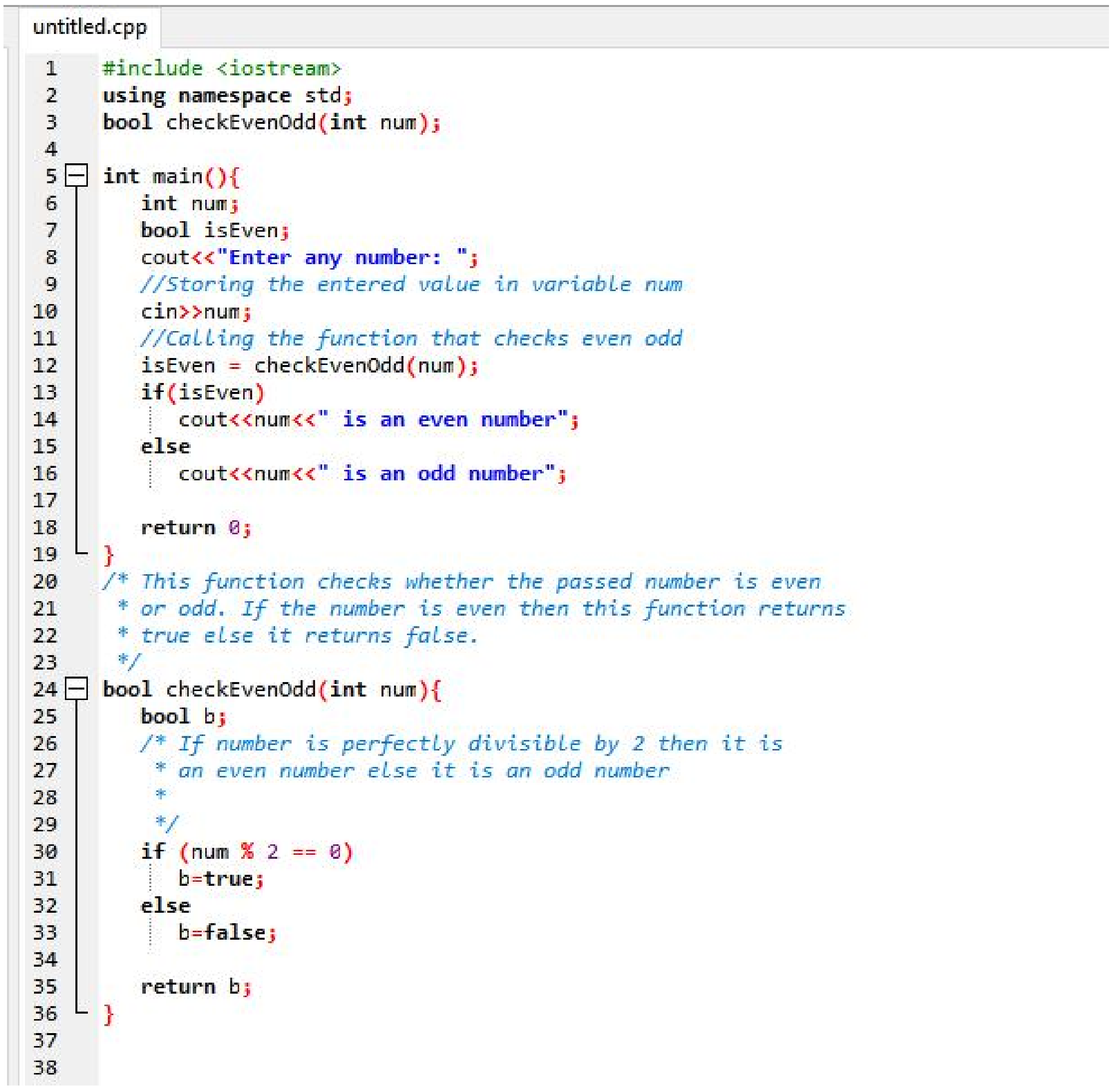
3) Do while loop:

It is exit controlled loop.
In this loop, the body of the loop is executed first
the condition check.

The syntax of the do-while loop is

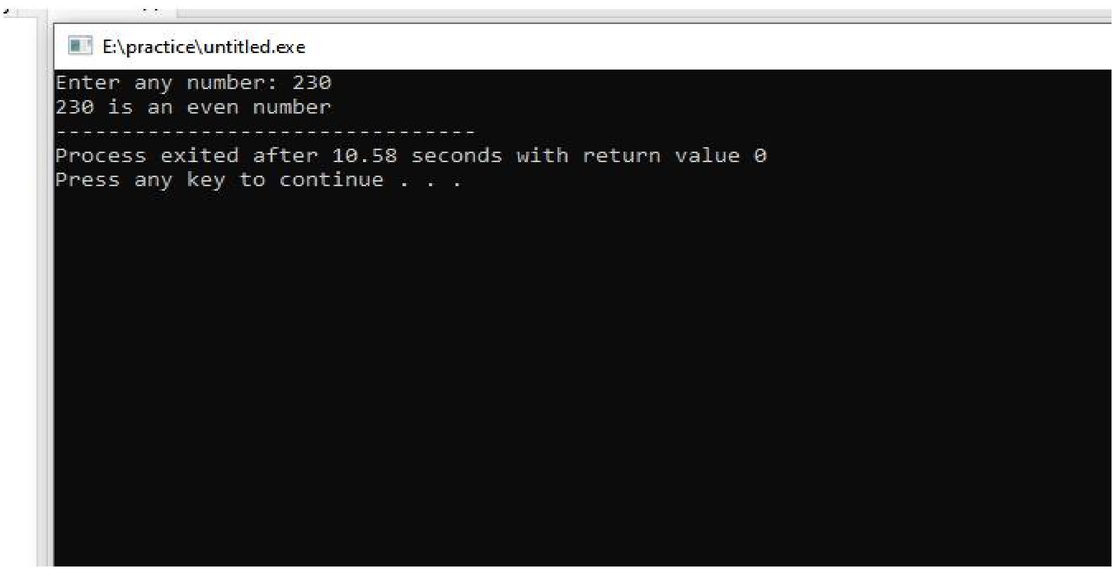
```
do
{
    //body of the loop
}while(condition)
```

Answer 3(b):



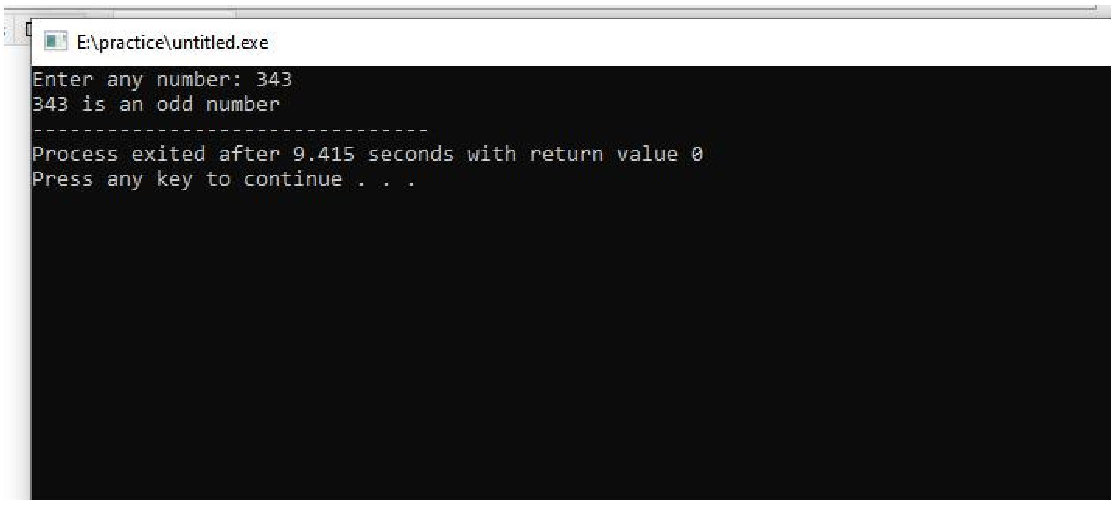
untitled.cpp

```cpp
#include <iostream>
using namespace std;
bool checkEvenOdd(int num);

int main(){
   int num;
   bool isEven;
   cout<<"Enter any number: ";
   //Storing the entered value in variable num
   cin>>num;
   //Calling the function that checks even odd
   isEven = checkEvenOdd(num);
   if(isEven)
      cout<<num<<" is an even number";
   else
      cout<<num<<" is an odd number";

   return 0;
}
/* This function checks whether the passed number is even
 * or odd. If the number is even then this function returns
 * true else it returns false.
 */
bool checkEvenOdd(int num){
   bool b;
   /* If number is perfectly divisible by 2 then it is
    * an even number else it is an odd number
    *
    */
   if (num % 2 == 0)
      b=true;
   else
      b=false;

   return b;
}
```



E:\practice\untitled.exe

```
Enter any number: 230
230 is an even number
--------------------------------
Process exited after 10.58 seconds with return value 0
Press any key to continue . . .
```



E:\practice\untitled.exe

```
Enter any number: 343
343 is an odd number
--------------------------------
Process exited after 9.415 seconds with return value 0
Press any key to continue . . .
```

Answer 4(a):

Purpose of “break” statement:

- When the “break” statement is located inside a loop, it will terminate the loop immediately. Then the program control resumes at the next statement following the loop.

- The “break” can be used to terminate a switch statement case.

- If the break statement used in the innermost loop of a nested loop, it will terminate the current loop and the program continues executing the next statement or immediate outer loop.

Purpose of “continue” statement:

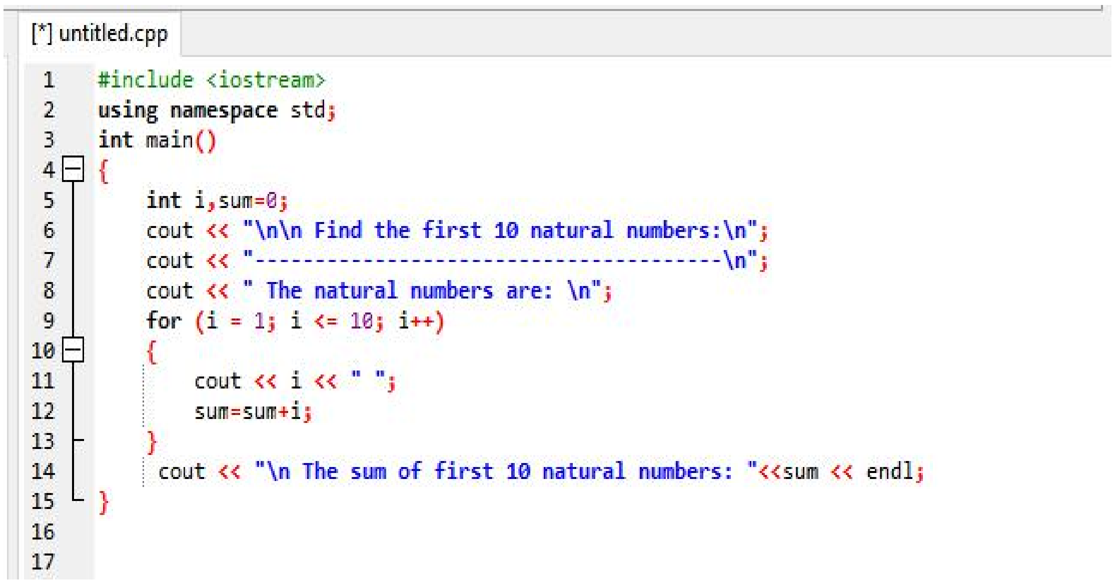
- It is used inside the loops.

- When a continue statement is encountered inside a loop, control jumps to the beginning of the loop for next iteration, skipping the execution of statements inside the body of loop fort the current iteration.

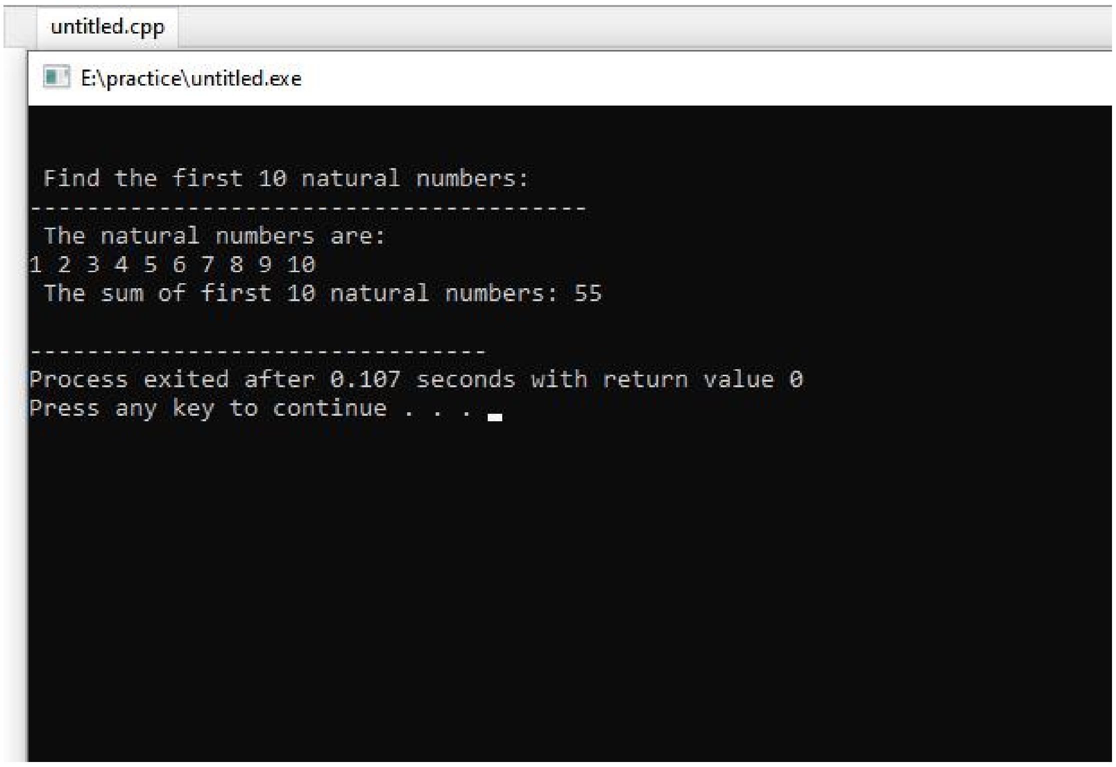
- In “for” loop, the “continue” statement causes the conditional test and increment portions of the loop to execute.

- In the “while” and “do-while” loops, “continue” statement causes the program control to pass to the conditional tests.

Answer 4(b):



```cpp
#include <iostream>
using namespace std;
int main()
{
    int i,sum=0;
    cout << "\n\n Find the first 10 natural numbers:\n";
    cout << "---------------------------------------\n";
    cout << " The natural numbers are: \n";
    for (i = 1; i <= 10; i++)
    {
        cout << i << " ";
        sum=sum+i;
    }
     cout << "\n The sum of first 10 natural numbers: "<<sum << endl;
}
```



```
 Find the first 10 natural numbers:
---------------------------------------
 The natural numbers are:
1 2 3 4 5 6 7 8 9 10
 The sum of first 10 natural numbers: 55

--------------------------------
Process exited after 0.107 seconds with return value 0
Press any key to continue . . .
```

Answer 5:

❖ Character set:
Character set is a set of valid characters that a language can recognize. A character represents any letter, digits, or any other sign.

Examples:

- Letters : A-Z, a-z
- Digits : 0-9
- Special Symbols : Space + - ∗ /^ \ ( ) [ ] { } = != < > . ′ ″ $, ; : % ! & _ # <= >= @
- White Spaces : Blank space, Horizontal tab (→), Carriage return (↵), Newline, Form feed
- Other Characters : C++ can process any of the 256 ASCII characters as data or as literals.

❖ Constants:
A constant, like a variable, is a memory location where a value can be stored. Unlike variables, constants never change in value.

Examples:

```
const int kill_bonus = 5000;

const int var = 5;
```

❖ Variable:
A variable is a name which is associated with a value that can be changed. For example when I write int num=20; here variable name is num which is associated with value 20, int is a data type that represents that this variable can hold integer values.

Examples:

```
a = 5;
b = 2;
a = a + 1;
result = a - b;
```

❖ Keyword:

Keyword is a predefined or reserved word in C++ library with a fixed meaning and used to perform an internal operation.

Keywords are those words whose meaning is already defined by Compiler. These keywords cannot be used as an identifier. Note that keywords are the collection of reserved words and predefined identifiers. Predefined identifiers are identifiers that are defined by the compiler but can be changed in meaning by the user.

Examples:

C++ provides 64 **keywords**

for, break, continue, switch, int float, double, char, try, catch, while, etc

❖ Relational Operators:

Relational operators are also known for comparison operators. Relational operators are used to relating the condition, that is it compares the two values and prints the result. In this article, we are going to see those relational operators in C++ with the help of examples.

Different Relational Operators in C++:

There are total 6 relational operators ==, !=, <, >,<=, >= which are explained below:

1. Less than Operator (<):

This operator is called less-than the operator. It checks whether the value of the left operand is less than the value of the right operand or not. If it satisfies the condition then, it returns true as a value else it returns false.

2. Greater than Operator (>):

This operator is called greater than the operator. It checks whether the value of the left operand is greater than the value of the right operand. If it satisfies the condition it returns true as value else it returns false.

3. Less than or Equal to Operator (<=):

This operator is called less than or equal to the operator. It checks whether the value of the left operand is less than or equal to the value of the right operand. If it satisfies the condition it returns true as value else it returns false.

4. Greater than or Equal to Operator (>=):

This operator is called as greater than or equal to the operator. It checks whether the value of the left operand is greater than or equal to the value of the right operand. If it satisfies the condition it returns true as value else it returns false.

5. Equal to Operator (==):

This operator is called as is equal to the operator. It checks whether the value of the left operand is equal to the value of the right operand. If it satisfies the condition it returns true as value else it returns false.

6. Not Equal to Operator (!=):

This operator is called as is not equal to the operator. It checks whether the value of the left operand is not equal to the value of the right operand. If it satisfies the condition it returns true as value else it returns false.

THE END