Chapter-5-Complete

C++: Finding Code Errors

5.4

a) Infinite Loop

Code: for (unsigned int x= 100; x>= 1; ++x)

Error: This is an infinite loop. The condition x>= 1 will always be true because x starts at 100 and increases indefinitely.

Correction: Change ++x to -- x to count down from 100 to 1.

b) Missing Break Statement

Code: switch (value % 2) {case 0: ... case 1: ...}

Error: Missing a break; statement after case 0. Without it, if the value is even, it will execute case 0 and then "fall through" to execute case 1 as well.

Correction: Add break; after the first cout statement.

c) Wrong Operator for Decrementing

Code: for (unsigned int x= 19; x>= 1; x += 2)

Error: To output odd integers from 19 down to 1, you must decrement. Using x += 2 will cause an infinite loop (or overflow) as x increases.

Correction: Change x += 2 to x -= 2.

d) Off-by-one / Logic Error

Code: do { ... counter += 2; } while (counter < 100);

Error: The loop stops before printing 100. When counter reaches 100, the condition 100 < 100 is false, so it exits without printing the final number. Also, "While" must be lowercase while.

Correction: Change the condition to while (counter <= 100) ;.

5.5 - Write a program that uses a for statement to sum a sequence of integers. Assume that the first integer specifies the number of values.

#include

using namespace std;

int main() {

int numberOfValues;

```
int currentInput;
int totalSum = 0;
cout << "Enter the number of integers to sum: ";
if (cin >> numberOfValues)
   cout << "Now enter " << numberOfValues << " values one by one:" << endl;
   for (int i = 1; i <= numberOfValues; ++i) {
      cin >> currentInput;
      totalSum += currentInput;
   }
   cout << "The sum of the values is: " << totalSum << endl;
} else {
   cout << "Invalid input for number of values." << endl;
}
return 0;
}
```

5.6 - Write a program that uses for statement to calculate the average of several integers.Assume the last sentinel 9999.

```
#include
using namespace std;
int main() {
int num, count = 0;
double sum = 0;
cout << "Enter integers (9999 to stop): " << endl;
for (cin >> num; num != 9999; cin >> num) {
   sum += num
   count++;
}
if (count != 0) {
    cout << "Total Sum = " << sum << endl;

    cout << "Total Count = " << count << endl;

    cout << "Average = " << sum / count << endl;

} else {

    cout << "No numbers were entered." << endl;

}

return 0;

}
```

5.7 - What does the following program do?

```
#include

using namespace

int main() {

unsigned int x; // declare x

unsigned int y; // declare y

cout << "Enter two integers in the range 1-20: ";

cin >> x >> y;

For (unsigned int i = 1; i <= y; ++i) // count from 1 to y

{

for (unsigned int j = 1; j <= x; ++j) // count from 1 to x

cout << '@'; // output @

cout << endl; // begin new line

}

}
```

Answer:

The program prints a rectangle of @ symbols with:

X columns

Y rows

Example

If I Input:

X = 5

Y = 3

Output:

```
@@@@@
@@@@@
@@@@@
```

This program uses nested for loops to display a rectangular pattern of @ symbols, where the number of columns and rows is determined by user input.

5.8 - Find the smallest integer ,write a program:

```
#include
using namespace std;
int main()
{
int n, num, smallest;
cout << “Enter number of values: “;
cin >> n;
cout << “Enter integers: “;
cin >> smallest; // assume first number is smallest
for (int i = 2; i <= n; i++)
{
cin >> num;
if (num < smallest)
Smallest = num;
}
cout << “Smallest integer is: “ << smallest << endl;
return 0; }
```

5.9 - Write a program that uses for statement to find the smallest of several integers.

```
#include
Using namespace std;
Int main()
```

```
{
int product = 1;
for (int i = 1; i <= 15; i += 2)
{
Product *= i;
}
cout << “Product of odd integers from 1 to 15 is: “ << product << endl;
return 0;
}
```

i+=2,as it increases by 2(1,3,...)

5.10 - Write a program to evaluate factorial of integers from 1 to 5?

```
#include
using namespace std;
int main()
{
long long factorial = 1;
cout << “Number\tFactorial\n”;
for (int i = 1; i <= 5; i++)
{
Factorial *= i;
cout << i << “\t” << factorial << endl;
}
return 0;
}
```

5.12 - Write a program (according to drawing patterns)

```
*
**
```

```
***

****

*****

******

*******

********

*********

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

Answer:

```
#include

using namespace std;

int main() {

// Outer loop for 10 rows

for (int i = 1; i <= 10; i++) {

// Inner loop prints stars equal to the row number

for (int j = 1; j <= i; j++) {

cout << “*”;

}

cout << endl; // Move to next line

}

return 0; }
```
(B) second part:

```
#include

using namespace std;

int main() {

// Outer loop starts at 10 and decreases to 1

for (int i = 10; i >= 1; i--)

{

for (int j = 1; j <= i; j++) {

cout << “*”;

}

cout << endl;
}
return 0; }
```

Output:

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

(C)Third part:

```
#include
using namespace std;
int main() {
for (int i = 10; i >= 1; i--) {
// Print spaces: Total width (10) minus stars
for (int s = 0; s < (10 – i); s++) {
cout << " ";
}
// Print stars
for (int j = 1; j <= i; j++) {
cout << "*";
}
cout << endl;
}
return 0; }
```

Output:

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

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

(D)4th part:

```
#include
using namespace std;
int main() {
for (int i = 1; i <= 10; i++) {
// Print spaces first
for (int s = 0; s < (10 – i); s++) {
cout << “ “;
}
// Print stars
for (int j = 1; j <= i; j++) {
cout << “*”;
}
cout << endl;
}
return 0; }
```

Output:

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

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

5.13 - Bar chart program:

```
#include
using namespace std;
int main() {
int number;
cout << “Enter 5 numbers between 1 and 30:” << endl;
for (int i = 1; i <= 5; i++) {
cin >> number;
// Inner loop prints the bar
for (int j = 1; j <= number; j++) {
cout << “*”;
}
cout << endl; // Move to next line for the next bar
}
return 0; }
```

Output:

```
Enter 5 numbers between 1 and 30
7
*******
15
***************
3
***
10
```

```
**********
5
*****
```

5.17 - What does each statement prints?

In C++, logical expressions evaluate to bool. When printed using cout, true is displayed as 1 and false is displayed as 0.

Based on the variables i=1, j=2, k=3, m=2:

Statement: cout << (i == 1) **Logic:** 1 == 1 is True 1

Statement: cout << (j == 3) **Logic:** 2 == 3 is False 0

Statement: cout << (i >= 1 && j < 4) **Logic:** 1 >=1 (T) AND 2 < 4 (T) is 1

Statement: cout << (m <= 99 && k < m) **Logic:** 2 <= 99 (T) AND 3 < 2 (F) 0

Statement: cout << (j >= i || k == m) **Logic:** 2 >= 1 (T) OR 3 == 2 (F)

5.21 - Demorgan's Law

Original Expression without De Morgan's law	Equivalent Expression using De Morgan's law
a) !(x < 5) && !(y >= 7)	!((x < 5) || (y >= 7))
b) !(a == b) || !(g != 5)	!((a == b) && (g != 5))
c) !((x <= 8) && (y > 4))	!(x <= 8) || !(y > 4)
d) !((i > 4) || (j <= 6))	!(i > 4) && !(j <= 6)

5.22 - write a program to show that the original and new expressions are equivalent.

```
#include
using namespace std;
int main() {
// Arbitrary values for testing
int x = 6, y = 5, a = 2, b = 3, g = 5, i = 5, j = 7;
// boolalpha se 1/0 ki jagah true/false print hota hai
cout << boolalpha;
// Test Case A: De Morgan's Law for AND
cout << "A: " << (!(x < 5) && !(y >= 7)) << " is equal to "
    << (!((x < 5) || (y >= 7))) << endl;
// Test Case B: De Morgan's Law for OR
cout << "B: " << (!(a == b) || !(g != 5))  << " is equal to "
    << (!((a == b) && (g != 5))) << endl;
// Test Case C: Negation of AND
cout << "C: " << (!((x <= 8) && (y > 4))) << " is equal to "
    << (!(x <= 8) || !(y > 4)) << endl;
// Test Case D: Negation of OR
cout << "D: " << (!((i > 4) || (j <= 6))) << " is equal to "
    << (!(i > 4) && !(j <= 6)) << endl;
return 0;
}
```

5.23 - Write a program that print the following diamond shape.

```
#include
using namespace std;
int main() {
int n = 5; // Diamond size
// 1. Upper Half
for (int i = 1; i <= n; i++) {
   // to print space
   for (int j = 1; j <= n - i; j++)
      cout << " ";
   }
   // to print stars
   for (int k = 1; k <= (2 * i - 1); k++) {
      cout << "*";
   }
```

```
    cout << endl;
}
// 2. Lower Half
for (int i = n - 1; i >= 1; i--) {
    // to print spaces
    for (int j = 1; j <= n - i; j++) {
        cout << " "
    }
    // to print star
    for (int k = 1; k <= (2 * i - 1); k++)
        cout << "*";
    }
    cout << endl;
}
return 0;
}
```

5.26 - What does this code do?

```
#include
using namespace std;
int main()
for (unsigned int i = 1; i <= 5; ++i) {
for (unsigned int j = 1; j <= 3; ++j) {
for (unsigned int k = 1; k <= 4; ++k) {
cout << ‘*’;
}
cout << endl;
}
cout << endl;
} return 0; }
```

Answer:

This program prints 5 separate blocks of asterisks, where each block consists of 3 rows and 4 columns of stars.

```
****

****

****


****

****

****


****

****

****


****

****

****


****

****

****
```