**Practical 7 – refer to Topics 13 and 14**

**Part A (Understanding Concepts)**

1. Consider the following function definition.

|  |  |  |
| --- | --- | --- |
| |  | | --- | | int f(int n)  {  int i, p;  p = 1;  for (i = 1; i <= n; i++)  p \*= i;  return p;  } | | |
| What is displayed by the following function calls? | |
| 1. cout << f(3) << endl; | 1\*2\*3 = 6 |
| 1. cout << f( f(3) ) << endl; | 1 x 2 x 3 x 4 x 5 x 6= 720 |

1. What type of loop would you use for the following programs: *counter-controlled*, *sentinel-controlled* or *general conditional* *loop*?

|  |  |
| --- | --- |
| 1. A program to compute the product of the numbers from 1 to 10. | Counter-controlled loop |
| 1. A program to compute the product of 10 numbers entered by the user. | Counter-controlled loop |
| 1. A program to compute the product of a set of numbers terminated by the sentinel number 0. | Sentinel-controlled loop |
| 1. A program to compute the product of a set of numbers. After each number is entered, the user is asked whether there are any more numbers to process. | General conditional loop |

1. Predict the output of the following program fragments, check your answer with the compiler and answer the related questions.

|  |  |  |
| --- | --- | --- |
| 1. for (i = 5; i >= 1; i--)   {  for (j = 1; j <= i; j++)  cout << setw(3) << i;  cout << endl;  } | | |
| What is the output? | 1  2 2  3 3 3  4 4 4 4  5 5 5 5 5 | |
| Modify the code to get the following output:  5 4 3 2 1  4 3 2 1  3 2 1  2 1  1 | for < (i = 5; i >= 1; i--)  {  for (j = i; j>=1; j--)  cout << setw(3) << j;  cout << endl;  } | |
| 1. for (i = 1; i <= 5; i++)   {  for (j = 1; j <= i; j++)  cout << setw(3) << i;  cout << endl;  } | | |
| What is the output? | 5 5 5 5 5  4 4 4 4  3 3 3  2 2  1 | |
| Modify the code to get the following output:  1  2 3  3 4 5  4 5 6 7  5 6 7 8 9 | for (i = 1 ; i <= 5; i++)  {  for(j = 1, k = i; j <= i; j++, k++)  cout << setw(3) << k;  cout << endl;  } | |
| (c)for (i = 5; i >= 1; i--)  {  for (k = 5-i; k >= 1; k--)  cout << " "; // print 3 spaces  for (j = 1; j <= i; j++)  cout << setw(3) << i;  cout << endl;  } | | |
| What is the output? | | 5 4 3 2 1  4 3 2 1  3 2 1  2 1  1 |
| Modify the code to get the following output:  1 1 1 1 1  2 2 2 2  3 3 3  4 4  5 | | for (i = 1; i <= 5; i++)  {  for (k = i -1; k >= 1; k--)  cout << “ “;  for (j = 1; j <= 6 - i; j++)  cout << setw(3) << i;  cout << endl;  } |
| (d)  for (int i = 5; i >= 1; i--)  {  for (int k = 5 - i; k >= 1; k--)  cout << “ “;  for (int j = 1; j <= i; j++)  cout << setw(3) << i;  cout << endl;  } | | 5 5 5 5 5  4 4 4 4  3 3 3  2 2  1 |

**Part B (Programming Exercises)**

1. Consider the following incomplete program to find the total and average of a set of numbers entered by a user.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | #include <iostream>  using namespace std;  int main(void)  {  int number; /\* number to be added \*/  int count; /\* count of numbers to be added \*/  int i; /\* loop counter variable \*/  int total; /\* total of the set of numbers \*/  double average; /\* average of the set of numbers \*/  total = 0;  cout << "How many numbers to process? ";  cin >> count;  for (i = 1; i <= count; i++)  {  cout << "Enter number " << i << " : ";  cin >> number;  total += number;  }      average = (double)total / count;  cout << "Total is " << total << endl;  cout << "Average is " << average << endl;  return 0;  } |

1. Add the following statements in the appropriate places in the program.
2. cout << "Total is " << total << endl;
3. cout << "Average is " << average << endl;
4. total = 0;
5. total += number;
6. average = (double)total / count;
7. Add a *do-while* loop in the *for* loop to validate the numbers entered. A number must be in the range from 1 to 100. If not, the program displays an error message before asking the user to enter the number again.

#include <iostream>

using namespace std;

int main(void)

{

int number; /\* number to be added \*/

int count; /\* count of numbers to be added \*/

int i; /\* loop counter variable \*/

int total; /\* total of the set of numbers \*/

double average; /\* average of the set of numbers \*/

total = 0;

cout << "How many numbers to process? ";

cin >> count;

total = 0;

for (i = 1; i <= count; i++)

{

do

{

cout << "Enter number " << i << " , ";

cin >> number;

if(number < 1 || number > 100)

cout << "Number must be between 1 to 100";

}

while (number < 1 || number > 100);

total += number;

}

average = (double)total / count;

cout << "Total is " << total << endl;

cout << "Average is " << average << endl;

return 0;

}

1. Write a program to compute the *product* of the numbers from 1 to 10.

#include <iostream>

using namespace std;

int main(void)

{

int product = 1;

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

{

product \*= i;

}

cout << "Product of number 1 to 10 is: " << product << endl;

return 0;

}

//product = product x 1\*2\*3\*4\*5...\*10

1. Write a program to compute the *product* of 10 numbers entered by the user.

#include <iostream>

using namespace std;

int main(void)

{

int i, product, number;

product = 1;

cout << "Enter 10 numbers to multiply\n";

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

{

cout << "Enter number " << i << ": ";

cin >> number;

product \*= number;

}

cout << "Product of the numbers is " << product << endl;

return 0;

}

1. Write a program to compute the *product* of a set of numbers terminated by the sentinel number -1.

#include <iostream>

using namespace std;

int main(void)

{

int product, number, input;

product = 1;

cout << "Enter numbers to multiply\n";

cout << "Press - 1 if no more numbers\n";

cout << "Enter first number : ";

cin >> number;

while (number != -1)

{

product \*= number;

cout << "Enter next number : ";

cin >> number;

}

cout << "Product of the numbers is " << product << endl;

return 0;

}

1. Write a program to compute the *product* of a set of numbers. After each number is entered, the user is asked whether there are any more numbers to process.

#include <iostream>

using namespace std;

int main(void)

{

int product, number;

char input;

do

{

product = 1;

cout << "Enter a number\n";

cin >> number;

product \*= number;

cout << "Any more numbers to process ? (y / n)";

cin >> input;

}

while (input == 'y' || input == 'Y');

cout << "Product of the numbers is " << product << endl;

return 0;

}

1. Write a program to determine the *product* of a set of numbers. The user is asked to enter control-z when there are no more numbers to enter. Use the EOF in the loop termination test.

#include <iostream>

using namespace std;

int main(void)

{

int product, number, status;

product = 1;

cout << "Enter the numbers to multiply.\n";

cout << "Press ctrl - z when no more numbers.\n";

cout << "Enter first number : ";

cin >> number;

while (cin)

{

product \*= number;

cout << "Enter a number: ";

cin >> number;

}

cout << "Product is: " << endl;

return 0;

}

1. Write a program to compute the *sum* of a sequence of numbers from a lower limit until an upper limit in increments of 1. The user will enter the lower and upper limits.

#include <iostream>

using namespace std;

int main(void)

{

int total = 0, lower, upper;

cout << "Enter the lower limit: " << endl;

cin >> lower;

cout << "Enter the upper limit: " << endl;

cin >> upper;

for (int i = lower; i <= upper; i++)

total += i;

cout << " Total of the number is : " << total << endl;

return 0;

1. Modify the program in question 7 to make sure that the lower limit is less than the upper limit. If not, the program will ask the user to enter them again until the limits entered are valid.

#include <iostream>

using namespace std;

int main(void)

{

int total = 0, lower, upper;

do

{

cout << "Enter the lower limit: " << endl;

cin >> lower;

cout << "Enter the upper limit: " << endl;

cin >> upper;

if (lower >= upper)

cout << " Lower limit must be less than upper limit" << endl;

}

while (lower >= upper);

for (int i = lower; i <= upper; i++)

total += i;

cout << " Sum of numbers from " << lower << " to " << upper << " is " << total << endl;

return 0;

}

1. Write a program to find the largest number among the 10 numbers entered by the user.

int I, maximum, number;

cout << “Enter 10 numbers\n”;

for ( i = 1 ; I <= 10 ; i++ )

{

cout << “Enter number “ << I << “: “;

cin >> number;

if (I == 1)

maximum = number;

else

if(maximum < number)

maximum = number;

}

cout << “The largest number is “ << maximum << endl;

return 0;

**Part C (Self-Review / Revision)**

1. What are the 3 actions that must be performed when computing a total of numbers in a loop? Where should the actions be placed with reference to the loop?
2. What are the 3 actions that must be performed when computing a product of numbers in a loop? Where should the actions be placed with reference to the loop?
3. What is a sentinel? What is a sentinel-controlled loop? What is the general format for this type of loop?
4. What is the statement in C++ for implementing a post-test loop?
5. What is a common use of a post-test loop?
6. What are nested loops?
7. What is a general conditional loop?

**Part D (Practice Exercises)**

1. Write a complete C++ program that prompts the user to enter a series of integers terminated by a 0. The program should then print:
2. The number of integers entered (not including the 0)
3. The average of the integers
4. The maximum integer
5. The minimum integer

Example:

Enter a series of integers terminated by a 0:

85 95 75 67 0

Output:

You have entered 4 integers.

The average is 80.5

The maximum is 95

The minimum is 67

#include <iostream>

using namespace std;

int main(void)

{

int num, max, min, sum = 0, count = 0;

double average;

cout << "Enter a series of integer terminated by a 0 " << endl;

cin >> num;

while (num != 0)

{

count++;

if (count = 1)

{

max = num;

min = num;

}

else

{

if (num > max)

max = num;

if (num < min)

min = num;

}

sum += num;

cin >> num;

}

average = (double)sum / count;

cout << "Average is " << average << endl;

cout << "Maximum is " << max << endl;

cout << "Minimum is " << min << endl;

return 0;

}

1. Write a program with the 2 sets of nested *for* statements to produce the following pattern:

1

Produced by first set of nested *for* statements

2 1

3 2 1

4 3 2 1

5 4 3 2 1

4 3 2 1

Produced by second set of nested *for* statements

3 2 1

2 1

1

#include <iostream>

#include <iomanip>

using namespace std;

int main(void)

{

for (int i = 1; i <= 5; i++)

{

for (int k = 5 - i;k >= 1; k--)

cout << " ";

for (int j = 1; j >= 1;j--)

cout << setw(3) << j;

cout << endl;

}

for (int i = 4; i >= 1; i--)

{

for (int k = 5 - i; k >= 1; k--)

cout << " ";

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

cout << setw(3) << j;

cout << endl;

}

return 0;

}