Repetition 6

REVIEW QUESTIONS

1. In a pretest loop, the limit test condition is tested first.

a. True

3. The value of a comma expression is the value of the first expression.

b. False

5. Which of the following statements about pretest loops is true?

e. Initialization must be done before the first execution of the loop.

7. Which of the following statements about loop updates is false?

b. Loop updates may be made before or after a loop iteration.

9. Which of the C++ loops is a pretest loop?

d. both the *for* and the *while*

11. Which of the following statements about *for* and *while* statements is false?

d. Both statements include initialization within the statement.

13. The __________ is not a jump statement.

d. case

15. The __________ standard measure of efficiency is considered the most efficient.

b. logarithmic

EXERCISES

17. All three loops print the set {12 11 10 9 8} one number to a line. (a) and (b) are pretest loops; () is a post-test loop.

19. Loops (a) and (b) print nothing because the limit test is false and they are pretest loops. Loop (c) prints 12 once because it is a post-test loop and therefore executes at least once.

21.

a.

```
x = 0;
do
    {
     cout >> x >> endl;
     x++;
    } while (x < 10);
```

b.

```
cin >> x;
if (x != 9999)
    do
         {
           cout << x << endl;
           cin  >> x;
         } while (x != 9999);
```

23.

a.

```
x = 1;
do
    {
      cout << x << endl;
      x++;
    } while (x < 100);
```

b.

```
if (cin >> x)
    {
      do
        {
          cout << x << endl;
        } while (cin >> x);
    } // if
```

25.

a.

```
for (x = 0; x < 100; x++)
    cout << x << endl;
```

b.

```
for (; cin >> x;)
    cout << x << endl;
```

27. Without greater detail on the specification as to what the output should be, we assume it is correct. It prints the following number series on separate lines:

```
    1  2  3  4  5  6  7  8  9  10
```

29.

a. On separate lines, it prints the following number series:

```
20  19  18  17  16  15  14  13  12  11  10
```

b. On separate lines, it prints the following number series:

```
20  18  16  14  12  10  8  6  4  2
```

31.

a. It prints:

```
      1
       2
        3
         4
```

```
    5
     6
      7
       8
        9
         10
          11
           12
            13
             14
              15
               16
                17
                 18
                  19
                   20
```

b. It prints:

```
                   20
                  19
                 18
                17
               16
              15
             14
            13
           12
          11
         10
        9
       8
      7
     6
    5
   4
  3
 2
1
```

33.

```
// A for statement that prints 60 asterisks

for (int i = 0; i < 60; i++)
    cout << '*';
```

PROBLEMS

35.

```
/* Calculate average of 'n' positive numbers.
      Written by:
      Date:
*/
#include <iostream>
#include <iomanip>
using namespace std;

int main ()
{
   cout << "How many numbers: ";
   int numOf;
   cin >> numOf;
```

```
    int     numCnt = 0;
    float   sum    = 0;
    float   avrg   = 0;
    float   numIn;
    cout.precision (2);
    cout << "Enter first number: ";

    for (int i = 0; i < numOf; i++)
       {
        cin >> numIn;
        if (numIn > 0)
           {
            sum += numIn;
            numCnt++;
            avrg = sum / numCnt;
           } // if > 0
        if (i < (numOf - 1))
           {
            cout << "Average: " << setw(6) << avrg;
            cout << " Enter next number " << setw(3)
                 << i + 2 << ": ";
           } // if
        else
            cout << "\n\nFinal Statistics\n";
       } // for

    cout << "Sum:      " << sum << endl;
    cout << "Invalid: " << numOf - numCnt << endl;
    cout << "Average: " << setw(6) << avrg << endl;

    return 0;
}   // main
```

37.

```
/*  Determine largest number in series and how many
    times it occurred in the series.
       Written by:
       Date:
*/
#include <iostream>
#include <iomanip>
#include <climits>
using namespace std;

int main ()
{
    cout << "How many numbers: ";
    int numOf;
    cin >> numOf;

    int   largest    = INT_MIN;
    int   numIn;
    int   cntLarge;
    cout.precision (2);

    for (int i = 0; i < numOf; i++)
       {
        cout << "Enter number: " << i + 1 << ": ";
        cin >> numIn;
        if (numIn > largest)
```

```
            {
             largest   = numIn;
             cntLarge = 1;
            } // if > largest
         else
            if (numIn == largest)
               cntLarge++;
        } // for

    cout << "Largest: " << largest << " occurred "
         << cntLarge << " times\n";

    return 0;
}   // main
```

39.

```
/*  ==================== pattern =================
    This function creates a pattern of '*'s given the
    height.
       Pre  Given height
       Post pattern printed
*/
void pattern (int height)
{
    for (int i = 1;  i <= height;  i++)
       {
        for (int j = 1;  j <= 13;  j++)
             cout << '*';
        cout << endl;
       } // for

    return;
}   // pattern
```

41.

```
/*  ================== pattern ==================
    This function creates a pattern given the height.
        Pre  Given height
        Post Pattern printed
*/
void pattern (int height)
{
    for (int i = 1;  i <= height;  i++)
         {
          for (int j = 1;  j < (i * 2);  j++)
               cout <<  '*';
          cout <<  endl;
         } // for
    return;
}   // pattern
```

43. Our solution handles both an even and odd number of rows.

```
/*  =============== pattern =====================
    This function creates a pattern given the height.
        Pre  Given height
        Post Pattern printed
```

```
*/
void pattern (int height)
{
   for (int i = 1;  i <= height / 2;  i++)
        {
         for (int j = 1;  j < (i * 2);  j++)
              cout << '*';
         cout << endl;
        } // for i <= height / 2   (first half)

   // Once again if height is odd
   if (height % 2)
        {
         for (int j = 1;  j <= height;  j++)
              cout << '*';
         cout << endl;
        }  // if height is odd


   for (int i = height / 2;  i >= 1;  i--)
        {
         for (int j = 1;  j < (i * 2);  j++)
              cout << '*';
         cout << endl;
        } // for i >= height / 2   (second half)
   return;
}  // pattern
```

45.

```
/*  Read integer data from standard input unit and print
    the number followed by the minimum integer read,
    maximum integer read, and the average of the list.
        Written by:
        Date:
*/
#include <iostream>
#include <climits>
using namespace std;

void minimum  (int& smallest, int current_no);
void maximum  (int& largest,  int current_no);
double average(int  sum,      int count);

int main ()
{
   cout << "\n*** start of prb0645.cpp ***\n\n";
   cout << "Please enter an integer :  ";
   int smallest = INT_MAX;
   int largest  = INT_MIN;
   int sum   = 0;
   int count = 0;
   int num;
   while  (!(cin >> num).eof())
      {
       minimum (smallest, num);
       maximum (largest,  num);
       sum += num;
       count++;
       cout << "Enter next integer <^Z> to stop : ";
      } // while
   cout << "\n\nThe minimum number is    :  "
        << smallest << endl;
   cout << "The maximum number is    :  "
        << largest  << endl;
   cout << "The average is           :  "
        << average(sum, count) << endl;
   cout << "\n***  end  of prb0645.cpp ***\n\n";
   return 0;
}  // main

/* ================= minimum ===================
   Compare 2 numbers and selects the smaller
       Pre  smallest is the current smallest number
            current_no is the contender
       Post smaller of the two is stored in 'smallest'
*/
void minimum (int& smallest, int  current_no)
{
   if (current_no < smallest)
        smallest = current_no;
   return;
}  // minimum

/* ================ maximum ==================
   Compare 2 numbers and selects the larger
       Pre  largest is the current largest number
            current_no is the contender
       Post larger of the two stored in 'largest'
*/
void maximum (int& largest, int  current_no)
{
   if (current_no > largest)
        largest = current_no;
   return;
}  // maximum

/* =============== average =================
   Returns the average of a set of numbers
       Pre  sum is total of all the numbers
            count is the number of numbers
       Post average returned
*/
double average(int sum, int count)
{
   double dblSum = sum;
   double ave    = dblSum / count;
   return ave;
}  // average
```

47.

```
/*  ============= allPositiveAvrg ==============
   Reads positive intgers and returns their average.
   If negative read, it returns a negative number.
       Pre  nothing
       Post data read--returns average or negative
*/
double allPositiveAvrg ()
{
   cout << "\nEnter a positive integer: ";
```

```
    int sum   = 0;
    int count = 0;
    int num;
    while (!(cin >> num).eof())
       {
        if (num < 0)
            return num;
        else
           {
            sum += num;
            count++;
            cout << "Next number or <^Z> to stop : ";
           } // else
       } // while
    return (static_cast<double>(sum) / count);
}   // allPositiveEOF
```

49.

```
/*  =============== smallestEOF ===================
    This program modifieds Program 6-20.  To initialize
    the smallest variable, read the first number and put
    its value in smallest, then go into the loop.
        Pre    Nothing
        Post   Series of numbers entered
               Smallest returned
*/
int smallestEOF ()
{
    int smallest;
    cout << "\nEnter an integer  :   ";
    cin  >> smallest;

    int num = smallest;
    do
       {
        if (num < smallest)
            smallest = num;
        cout << "Next integer (<^Z> to quit)  :  ";
       } while (!(cin >> num).eof());
    return smallest;
}   // smallestEOF
```

51.

```
/*  Read an integer from the keyboard and then call a
    recursive function to print it out in reverse.
        Written by:
        Date:
*/
#include <iostream>
using namespace std;

void print_reversed (long num);

int main ()
{
    cout << "\n*** start of prb0651.cpp ***\n\n";

    cout << "\nEnter an integer  :   ";
    long number;
```

```
   cin  >> number;

   cout << "\nThe number reversed is  :   ";
   print_reversed (number);

   cout << "\n\n***  end  of prb0651.cpp ***\n\n";

   return 0;
}  // main

/* ============= print_reversed ================
   Print integer in reverse order using recursion.
       Pre  original contains number to be reversed
       Post number printed in reverse order
*/
void print_reversed (long num)
{
   if (num  !=  0)
      {
       cout << (num % 10);
       print_reversed (num / 10);
      } // if num != 0
   return;
}  // print_reversed
```