Midterm Exam Answers
Introduction to Scientific Computing with C++

Name: **30 June 2011**

- *Answer TEN of the following ELEVEN Questions. Cross out the number of the question you wish to omit.*
- *During this exam, you are allowed to refer to notes you have made, on two sheets of paper. No other references are allowed.*
- *I am not asking you to write complete C++ programs! However, when you write your C++ statements, you should include the necessary declarations and initializations.*
- *If I ask you to write a C++ function, I want a complete function, including the function header, curly brackets, and returned value.*

---

1. Write a **for** loop which prints the integers from 100 down to 0 by fives; that is, the first three lines should be 100, 95, 90...

```
int i;

for ( i = 100; 0 <= i; i = i - 5 )
{
  cout << i << "\n";
}
```

---

2. A user is going to input several integers to a program, terminating with an extra, dummy value of 999. Write C++ code which reads the user's data and computes the sum, while ignoring the final 999 value.

```
int i, sum = 0;

while ( true )
{
  cin >> i;
  if ( i == 999 )
  {
    break;
```

```
  }
  sum = sum + i;
}
```

---

3. A cab company charges $2.50 on entry into the cab, an additional $1.75 per mile for each of the first 10 miles, and an additional $1.00 per mile for each mile traveled beyond 10 miles. If the **int** variable **miles** contains the miles traveled, display the C++ statements necessary to compute **bill**, the amount of money owed.

```
int miles;
float bill;

if ( miles < 10 )
{
  bill = 2.50 + 1.75 * ( double ) * miles;
}
else
{
  bill = 2.50 + 1.75 * 10.0 + 1.00 * ( double ) ( miles - 10 );
}
```

---

4. Write C++ code that will find and print the first integer which is greater than 1000, is divisible by 347, and is not divisible by 7.

```
int i = 1000;

while ( true )
{
  i = i + 1;
  if ( ( i % 347 == 0 ) && ( i % 7 ) != 0 ) )
  {
    break;
  }
}
```

---

5. The array **a** has dimension 100. Write C++ code that will print the array using 25 lines of output, with each line containing four entries of the array.

```
int a[100], i;

for ( i = 0; i < 100; i = i + 4 )
{
  cout << "  " << a[i]
       << "  " << a[i+1]
       << "  " << a[i+2]
       << "  " << a[i+3]
       << "\n";
}
```

---

6. Suppose the function **rand_float()** has returned a random number **r** between 0 and 1. Write a one-line C++ formula that converts **r** to a random number **s** between the values **a** and **b**.

```
float a, b, r, s;

r = random_float ( );
s = a + ( b - a ) * r;
```

---

7. Suppose the variable **c** has been declared as **double c[100]**, and that values have been assigned to each entry, and that many, but not all, values are zero. Write C++ code that will find and print the largest index **i** such that **c[i]** is not zero.

```
double c[100];
int i;

for ( i = 99; 0 <= i; i-- )
{
  if ( c[i] != 0.0 )
  {
    cout << "c[" << i << "] is not zero.\n";
    break;
  }
}
```

---

8. Write a C++ function called **summer()** that accepts a **float** array **d** and an integer **n**, the dimension of the array, and which returns the number of negative entries in **d**.

```
int summer ( float d[], int n )
{
  int i, value = 0;

  for ( i = 0; i < n; i++ )
  {
    if ( d[i] < 0.0 )
    {
      value = value + 1;
    }
  }
  return value;
}
```

---

9. A sequence starts with $x_0 = 0.5$, and subsequent entries are determined by:

$$x_{n+1} = 0.7 * x_n * (1 - x_n);$$

Write a C++ loop that computes the value of the 1,000,000th entry, but which uses as little memory as possible. In other words, do not use an array to store the entries of **x**!

```
  double xn, xnp1;
  int i;

  xnp1 = 0.5;
  for ( i = 1; i <= 1000000; i++ )
  {
    xn = xnp1;
    xnp1 = 0.7 * xn * ( 1.0 - xn );
  }
```

---

10. The factorial function n! is defined by

$$n! \equiv n \cdot n - 1 \cdot ... \cdot 3 \cdot 2 \cdot 1$$

and 0! is defined to be 1. Write a C++ function called **fact()** whose input is a nonnegative integer **n** and which returns the value of **n!**.

```
int fact ( int n )
{
```

```
  int i, value = 1;

  for ( i = 1; i <= n; i++ )
  {
    value = value * i;
  }
  return value;
}
```

---

11. Write C++ statements to estimate the integral of $f(x) = x^2 + \sin(x)$ over the interval [0,1], using 10,000 function evaluations. You may assume that there is a function available, called **random_double()**, which will return random **double** values in the interval [0,1].

```
  double fx, integral, sum, x;
  int i;

  sum = 0.0;
  for ( i = 1; i <= 10000; i++ )
  {
    x = random_double ( );
    fx = x * x + sin ( x );
    sum = sum + fx;
  }
  integral = 1.0 * sum / 10000;
```