Functions (continued)

Chapter 6

Passing Arguments

- Pass by value
  - arguments are **copied** into the parameter list
  - changes made in the function will **not** be reflected in main
- Pass by reference
  - changes made in the function are reflected in the main

Example

```
void ValTest(int parm1, int parm2)
{
    parm1 = 33;
    parm2 = 44;
}

void RefTest(int &parm1, int &parm2)
{
    parm1 = 77;
    parm2 = 88;
}

int main()
{
    int val1 = 0, val2 = 0, val3 = 0, val4 = 0;

    ValTest(val1, val2);
    cout << "val1 = " << val1 << ", val2 = " << val2 << endl;

    RefTest(val3, val4);
    cout << "val3 = " << val3 << ", val4 = " << val4 << endl;

    return(0);
}
```

Example

```
void swap(int & num1, int & num2);
int main()
{
  int i, j;
  cin >> i >> j;
  swap(i,j);
  cout << i << j;
  return 0;
}

void swap(int & num1, int & num2)
{
  int temp;
  temp = num1;
  num1 = num2;
  num2 = temp;
  return;
}
```

What is the output?

```
void changeIt(int, int&, int&);
int main()
{
  int i, j, k, l;
  i = 2;
  j = 3;
  k = 4;
  l = 5;
  changeIt(i, j, k);
  cout << i << j << k << endl;
  changeIt(k, l, i);
  cout << i << k << l << endl;
}
```

```
void changeIt(int j, int&
  i, int& l)
{
  i++;
  j += 2;
  l += i;
}
```

Rules for Parameter Lists

- Same number of arguments as parameters
- Arguments & parameters are matched by position
- Arguments & parameters must have the same type
- The names of the arguments and parameters may be the same or different
- For reference parameters only, the parameter must be a single, simple variable

Example

- Given the following function prototype:

```
void checkIt(float &, float &, int, int, char &);
```

- And declarations in main:

```
float x, y;

int m;

char next;
```

Which are legal?

```
checkIt(x, y, m+3, 10, next);
checkIt(m, x, 30, 10, 'c');
checkIt(x, y, m, 10);
checkIt(35.0, y, m, 12, next);
checkIt(x, y, m, m, c);
```

Program

- Write a function to compute the median and average of three integers, and return the two values.
- An example function call would look like:
  - **`medianAndAverage(4, 5, 6, median, average);`**

bool return values

```
bool isEven (int value)
{
  return (value % 2) == 0;
}

int main()
{
  int x = 9, y = 10;
  if( isEven(x) )
  {
    cout << "EVEN: " << x << endl;
  }
  if( isEven(y) )
  {
    cout << "EVEN: " << y << endl;
  }
  return 0;
}
```

Practice

- Write a function, isVowel() that accepts a character and produces a boolean
  - isDigit()
  - isConsonant()
  - isWhitespace()
- Use these functions to count the number of vowels, consonants, whitespace, and digits in the file “gibberish.txt”

Practice

- The Pochhammer symbol (or *falling factorial*) $(x)_n$ is used to represent x! / ((x - n)!), where z! is z factorial
- Write a function to calculate factorial.
- Using the above function, write a function to calculate the falling factorial

Return types? Parameters?

Practice

- Use the previous functions to build a function that, given a maximum value for n, will approximate Pi

$$\pi = \frac{2\sqrt{3}}{Z} \quad Z = \sum_{n=0}^{\infty} \frac{(8n+1)\left(\frac{1}{2}\right)_n \left(\frac{1}{4}\right)_n \left(\frac{3}{4}\right)_n}{(n!)^3 9^n}$$

- cmath contains

```
double pow (double base, double exponent);
double sqrt (double x);
```

http://en.wikipedia.org/wiki/Pi

Practice

- Write a function to calculate the area of a rectangle. This function should produce a value and return it to the calling function.
- Write another function to calculate the area of a circle.
  - what data type should each function return?
  - what parameters should each function accept?

Practice

- Build a small program that asks the user for either a rectangle or circle and displays the area of the selection shape. Use the functions we just defined.
- Continue asking for input until the user types something other than ‘r’ or ‘c’.