Functions & Parameter Passing

Last Time

- We
  - Continued our discussion of functions
- Today we will
  - Learn about parameter passing mechanisms, specifically, passing by value versus passing by reference

Example

```c
void larger(int, int, int&);
int main()
{
  int num1, num2, large;
  cin >> num1 >> num2;
  larger(num1, num2, large);
  cout << "Larger number is " << large << endl;
}
void larger(int num1, int num2, int & large)
{
  if (num1 > num2)
    large = num1;
  else
    large = num2;
}
```

Passing Arguments

- Pass by value
  - Values are passed into the function
  - Any changes made in the function are not reflected in the main
- Pass by reference
  - Any changes made in the function are reflected in the main

Example

```c
void swap(int &, int &);
int main(void)
{
  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;
}
```

Rules for Parameter Lists

- There must be the same number of actual and formal arguments
- The correspondence between actual and formal arguments is by position only
- Corresponding actual and formal arguments must match in type
- The names of the actual and formal arguments may be the same or different
- For reference arguments only, the actual argument must be a single, simple variable
Example

- Given the following function prototype:
  
  ```c
  void checkIt(float &, float &, int, int, char &);
  ```

- And declarations in main:
  
  ```c
  float x, y;
  int m;
  char next;
  ```

- 16.1: Which are legal calls to function `checkIt`?

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

- 16.2: What is the output?

  ```c
  void changeIt (int, int, int &);
  void 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;
    changeIt(i, j, j);
    cout << i << j << endl;
    changeIt(i, i, i);
    cout << i << endl;
  }
  ```

- 16.3: What is the output?

  ```c
  void changeIt(int, int &, int &);
  void 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;
  }
  ```

- 16.4: Write a function to compute the sum and average of two integers, and return the values of sum and average.

  - An example function call would look like:
    ```c
    compute (4, 5, sum, average);
    ```

Summary

- In today’s lecture we covered
  
  - Library functions
  - Programmer defined functions

- Readings
  
  - P. 170 - 180