3 Input and Output

- A **data stream** is a sequence of data
  - Typically in the form of characters or numbers
- An input stream is data for the program to use
  - Typically originates
    - at the keyboard
    - at a file
- An output stream is the program's output
  - Destination is typically
    - the monitor
    - a file

3.1 Output using "cout"

- **cout** is an output stream sending data to the monitor
- The insertion operator "<<" inserts data into **cout**
- Example: `cout << number_of_bars << " candy bars\n";`
  - This line sends two items to the monitor
    - The value of **number_of_bars**
    - The quoted string of characters **" candy bars\n"**
      - Notice the space before the '**c**' in candy
      - The '**\n**' causes a new line to be started following the '**s**' in bars
    - A new insertion operator is used for each item of output

3.1.1 Examples Using cout

- This produces the same result as the previous sample
```
cout << number_of_bars;
cout << " candy bars\n";
```
- Here arithmetic is performed in the **cout** statement
```
cout << "Total cost is $" << (price + tax);
```
- Quoted strings are enclosed in double quotes ("Walter")
  - Don't use two single quotes (')
- A blank space can also be inserted with
```
cout << " ";
```
  if there are no strings in which a space is desired as in **" candy bars\n"**

3.2 Escape Sequences

- Escape sequences tell the compiler to treat characters in a special way
- '\' is the escape character
  - To create a newline in output use \n

    ```
    cout << "\n";
    ```

    or the newer alternative

    ```
    cout << endl;
    ```

  - Other escape sequences:
    * \t – a tab
    * \\\\ – a backslash character
    * \\" – a quote character

3.3 Formatting Real Numbers

- Real numbers (type **double**) produce a variety of outputs

  ```
  double price = 78.5;
  cout << "The price is $" << price << endl;
  ```

- The output could be any of these:
  The price is \$78.5
  The price is \$78.500000
  The price is \$7.850000e01
- The most unlikely output is:
  The price is \$78.50

3.3.1 Showing Decimal Places

- **cout** includes tools to specify the output of type double
- To specify fixed point notation
  - **setf**(ios::fixed)
- To specify that the decimal point will always be shown
  - **setf**(ios::showpoint)
- To specify that two decimal places will always be shown
  - **precision**(2)
- Example:

  ```
  cout.setf(ios::fixed);
  cout.setf(ios::showpoint);
  cout.precision(2);
  cout << "The price is " << price << endl;
  ```

3.4 Input Using "cin"

- cin is an input stream bringing data from the keyboard
- The extraction operator (>>) removes data to be used
- Example:

```
cout << "Enter the number of bars in a package\n";
cout << " and the weight in ounces of one bar.\n";
cin >> number_of_bars; cin >> one_weight;
```

- This code prompts the user to enter data then reads two data items from **cin**
  - The first value read is stored in number_of_bars
  - The second value read is stored in one_weight
  - Data is separated by spaces when entered

3.4.1 Reading Data From cin

- Multiple data items are separated by spaces
- Data is not read until the enter key is pressed
  - Allows user to make corrections
- Example:

```
cin >> v1 >> v2 >> v3;
```

  - Requires three space separated values
  - User might type
    34 45 12 <enter key>

3.5 Designing Input and Output

- Prompt the user for input that is desired
  - **cout** statements provide instructions

    ```
    cout << "Enter your age: ";
    cin >> age;
    ```

    * Notice the absence of a new line before using **cin**
- Echo the input by displaying what was read
  - Gives the user a chance to verify data

    ```
    cout << age << " was entered." << endl;
    ```

3.5.1 Sample02

```
//
//  sample02
//
//  Created by Hideki Fujioka on 3/12/13.
//  Copyright (c) 2013 Tulane University. All rights reserved.
//

#include <iostream>

using namespace std;

int main(int argc, const char * argv[])
{
    int a, b;

    cout << "input two numbers :";
    cin >> a >> b;

    cout << "a=" << a << " b=" << b << endl;

    return 0;
}
```