Input / Output (I/O)

- I/O streams
- Interactive I/O
- `cin >>`
- `cout <<`
- `get`
- `getline`
Input and Output

- I/O operations act on streams
  - Infinite (conceptually) sequences of data
  - Standard I/O streams comprise ASCII characters
Standard I/O Streams

Used for interactive I/O

- **cout**
  - Standard output stream
  - Outputs to default output device (monitor)
Standard I/O Streams

Used for interactive I/O

- **cout**
  - Standard output stream
  - Outputs to default output device (monitor)

- **cin**
  - Standard input stream
  - Input from default input device (keyboard)
  - Typically “echoed” on default output device
Interactive I/O using iostream library

#include <iostream>

input data
Keyboard

executing program

output data
Monitor

cin

cout
Example: cout <<

cout << "One" << "Two" << "Three";
Example: cout <<

cout << "One" << "Two" << "Three";

OneTwoThree
cout << "One" << "Two" << "Three";

OneTwoThree

cout << "One"
   << "Two"
   << "Three";
Example: `cout` <<

```cpp
cout << "One" << "Two" << "Three";
```

OneTwoThree

```cpp
cout << "One"
    << "Two"
    << "Three";
```

OneTwoThree
cout << "One" << "Two" << "Three";

OneTwoThree

cout
   <<
   "One"
      << "Two" <<
   "Three" ;
Example: cout <<

```cpp
cout << "One" << "Two" << "Three";
OneTwoThree

cout << "One" "Two" "Three";
OneTwoThree
```

```cpp
OneTwoThree
```
cout << "One" << "Two" << "Three";

OneTwoThree

cout << "One"
    << "Two"
    << "Three";

OneTwoThree
Example: `cout <<`

cout << "One" << "Two" << "Three";

OneTwoThree

cout << "One"
    << "Two"
    << "Three";

OneTwoThree

cout << "One" << " Two" << " Three";
Example: cout <<

cout << "One" << "Two" << "Three";

OneTwoThree

cout << "One"
   << "Two"
   << "Three";

OneTwoThree

cout << "One" << " Two" << " Three";

One Two Three
Example: `cout <<`
cout << "Hello there world!"; 

Hello there world!
cout << "Hello there world!";

Hello there world!

cout << "Hello there world!"
    << "C++ is amusing.";
Example: `cout <<` 

cout << "Hello there world!";

Hello there world!

cout << "Hello there world!"
   << "C++ is amusing.";

Hello there world!C++ is amusing.
cout << "Hello there world!";
Hello there world!

cout << "Hello there world!" << "C++ is amusing.";
Hello there world!C++ is amusing.

cout << "Hello there world!" << endl << "C++ is amusing.";
cout << "Hello there world!";
Hello there world!

cout << "Hello there world!"
    << "C++ is amusing.";
Hello there world!C++ is amusing.

cout << "Hello there world!" << endl
    << "C++ is amusing.";
Hello there world!
C++ is amusing.
Example: `cout <<

cout << "Hello there!" << endl << endl << "Bye";`
Example: cout <<

```cpp
cout << "Hello there!" << endl << endl << "Bye";
```

Hello there!

Bye
# Special Output Characters

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\n</td>
<td>new line</td>
</tr>
<tr>
<td>\t</td>
<td>tab</td>
</tr>
<tr>
<td>\b</td>
<td>backspace</td>
</tr>
<tr>
<td>\r</td>
<td>carriage return</td>
</tr>
<tr>
<td>'</td>
<td>single quote</td>
</tr>
<tr>
<td>&quot;</td>
<td>double quote</td>
</tr>
<tr>
<td>\</td>
<td>backslash</td>
</tr>
</tbody>
</table>
cout << "Hello \t and \n goodbye!";
cout << "Hello \t and \n goodbye!";

Hello and goodbye!
#include <iostream>
using namespace std;

int main()
{
    int length, width;
    cout << "Enter the length: ";
    cin >> length;

    cout << "Enter the width: ";
    cin >> width;

    cout << "The area of the rectangle is: "
         << length * width << endl;
    return 0;
}
#include <iostream>
using namespace std;

int main()
{
    int length, width;
    cout << "Enter the length: 
" << cin >> length;
    cout << "Enter the width: 
" << cin >> width;
    cout << "The area of the rectangle is: 
" << length * width << endl;
    return 0;
}
#include <iostream>
using namespace std;

int main()
{

    cout << "Enter the length: ";

    cout << "Enter the width: ";

    return 0;
}
#include <iostream>
using namespace std;

int main()
{
    cout << "Enter the length: ";
    cin >> length;
    cout << "Enter the width: ";
    cin >> width;
    cout << "The area of the rectangle is: " << length * width << endl;
    return 0;
}
#include <iostream>
using namespace std;

int main()
{
    cout << "Enter the length: ";
    cin >> length;

    cout << "Enter the width: ";
    cin >> width;

    return 0;
}
```cpp
#include <iostream>
using namespace std;

int main()
{
    int length, width;
    cout << "Enter the length: ";
    cin >> length;

    cout << "Enter the width: ";
    cin >> width;

    return 0;
}
```
#include <iostream>
using namespace std;

int main()
{
    int length, width;
    cout << "Enter the length: ";
    cin >> length;

    cout << "Enter the width: ";
    cin >> width;

    cout << "The area of the rectangle is: " 
         << length * width << endl;
    return 0;
}
#include <iostream>
using namespace std;

int main()
{
    int length, width;
    cout << "Enter the length: ";
    cin >> length;

    cout << "Enter the width: ";
    cin >> width;

    cout << "The area of the rectangle is: " << length * width << endl;
    return 0;
}
Enter the length: □
Enter the length: □

□ Indicates the cursor position where the next character will be printed
Contents of the output window

Enter the length: □
Enter the length: 10 <ENTER>
Enter the length: 10 <ENTER>
Enter the width: 0
Enter the length: 10 <ENTER>
Enter the width: 20 <ENTER>
Enter the length: 10 <ENTER>
Enter the width: 20 <ENTER>
The area of the rectangle is: 200
Useful Experiment

- Take this correctly working program
  - Experiment entering various types of invalid data
  - Experiment changing the program slightly
  - Observe what happens
  - You can learn a lot this way
Leading white space characters include:
- blanks, tabs, and newlines

Extraction operator $\gg$
- "Skips over" leading white space characters
- As it reads data from the input stream
- Actually reads but does not store anywhere
Interactive Input: cin >>

```cpp
int age;
cout << "age:";
cin >> age;
```
int age;
cout << "age:";
cin >> age;
```cpp
int age;
cout << "age:";
cin >> age;
```

```
age:25<ENTER>
```
```cpp
int age;
cout << "age:";
cin >> age; // age ← 25
age:25<ENTER>
```
int age;
cout << "age:";
cin >> age;

age: 25 <ENTER>
int age;
cout << "age:";
cin >> age;     // age ← 25

age: 25 <ENTER>
Interactive Input: cin >>

```cpp
int age;
cout << "age:";
cin >> age;
```

```
age: <ENTER>
<ENTER>
<ENTER>
25<ENTER>
```
Interactive Input: cin >>

```cpp
int age;
cout << "age:";
cin >> age;  // age ← 25
```

**age:**<ENTER>
<ENTER>
<ENTER>
25  <ENTER>
Interactive Input: cin >>

```cpp
int i;  char c;  double d;

cin >> i;
cin >> c;
cin >> d;
```
int i; char c; double d;

cin >> i;
cin >> c;
cin >> d;

25 J 2<ENTER>
int i; char c; double d;

    cin >> i;
    cin >> c;
    cin >> d;

25  J  2<ENTER>

_ indicates the position of the file reading marker
Interactive Input: cin >>

```cpp
int i;  char c;  double d;

cin >> i;
cin >> c;
cin >> d;
```

25 J 2<ENTER>
Interactive Input: cin >>

```cpp
int i; char c; double d;

cin >> i;       // i ← 25
cin >> c;
cin >> d;
```

`25 J 2<ENTER>"
int i;  char c;  double d;

```cpp
cin >> i;     // i ← 25
cin >> c;     // c ← 'J'
cin >> d;
```

```
25  J 2<ENTER>
```
int i;  char c;  double d;

```cpp
    cin >> i;  // i ← 25
    cin >> c;  // c ← 'J'
    cin >> d;  // d ← 2.0
```
int i;  char c;  double d;

cin >> i;
cin >> c;
cin >> d;
int i; char c; double d;

```cpp
    cin >> i;
    cin >> c;
    cin >> d;
```

25 A<ENTER>  
16.9<ENTER>
int i; char c; double d;

cin >> i; // i ← 25

25 A<ENTER>

cin >> c;

16.9<ENTER>

cin >> d;

16.9<ENTER>
Interactive Input: cin >>

```cpp
int i; char c; double d;

cin >> i; // i ← 25
cin >> c; // c ← 'A'
cin >> d;
```

25 A<ENTER>
16.9<ENTER>
Interactive Input: cin >>

```cpp
int i;  char c;  double d;

cin >> i;  // i ← 25
cin >> c;  // c ← 'A'
cin >> d;  // d ← 16.9
```
Interactive Input: cin >>

```cpp
int i; char c; double d;

cin >> i;
cin >> c;
cin >> d;
```
int i; char c; double d;

```c
int i;
cin >> i;
cin >> c;
cin >> d;
```

```
25 16.9A<ENTER>
```
Interactive Input: cin >>

```cpp
int i; char c; double d;

cin >> i;  // i ← 25
cin >> c;
 cin >> d;
```

```
25_16.9A<ENTER>
```
int i;  char c;  double d;

cin >> i;  // i ← 25
cin >> c;  // c ← ‘1’
cin >> d;
Interactive Input: cin >>

```cpp
int i;  char c;  double d;

cin >> i;    // i ← 25
cin >> c;    // c ← '1'
cin >> d;    // d ← 6.9
```

```
25 16.9A<ENTER>
```
int i; char c; double d;

cin >> i;
cin >> c;
cin >> d;
int i; char c; double d;

```cpp
    cin >> i;
cin >> c;
cin >> d;
```

```
25 1A<ENTER>
```
Interactive Input: cin >>

```cpp
int i; char c; double d;

cin >> i;    // i ← 25
cin >> c;
 cin >> d;
```

```
25_1A<ENTER>
```
int i;  char c;  double d;

cin >> i;    // i ← 25
cin >> c;    // c ← '1'
cin >> d;
Interactive Input: cin >>

```cpp
int i; char c; double d;

cin >> i;   // i ← 25
cin >> c;   // c ← '1'
cin >> d;   // cin enters a fail state
            // nothing is read into d
```

```
25 1A<ENTER>
```
cin enters *fail state*

- When datatype of value read doesn't match datatype of variable
cin enters **fail state**

- When datatype of value read doesn't match datatype of variable
- NO reading takes place again until cleared
cin enters \textit{fail state}

- When datatype of value read doesn't match datatype of variable
- NO reading takes place again until cleared
- More LATER
get()

- Get's one character
- Reads the next char **without** skipping whitespace
get()

char first, middle, last;

cin.get(first);
cin.get(middle);
cin.get(last);

A B C<ENTER>
get()

```cpp
char first, middle, last;

cin.get(first);   // first ← 'A'
cin.get(middle);  // middle ← ' '
cin.get(last);    // last ← 'B'
```

A B C<ENTER>
#include <string>
string firstName, lastName;

cin >> firstName;

Joe Smith 23
#include <string>

string firstName, lastName;

cin >> firstName;  // firstName ← “Joe”
cin >> lastName;   // lastName ← “Smith”

Joe Smith_23<ENTER>
Extraction operator >>

As always:

- >> operator **skips any leading whitespace**
- Reads successive char's into the string
- **Stops at the first trailing whitespace**
- Which is not consumed, but remains in input stream
getline ( <stream>, <stringvar> );

- Leading whitespace NOT skipped
- Reading stops when newline(\n) encountered
  - Read pointer after \n
- \n not put into string
getline( ) Function

- `getline ( <stream>, <stringvar>);`

- Leading whitespace NOT skipped
- Reading stops when newline(`\n`) encountered
  - Read pointer after `<\n>`
  - `<\n>` not put into string

```cpp
string message;
getline(cin, message);
```
#include <string>
string firstName, lastName;

ggetline(cin, firstName);
ggetline(cin, lastName);

Joe Smith 23
46 Bananas
Apple
#include <string>
string firstName, lastName;

g++line(cin, firstName); // firstName ← “Joe Smith 23”
g++line(cin, lastName);  // lastName ← “46 Bananas”
cin >> in combo with getline

Code:

```cpp
int x;
string restOfLine;
string name;
cout << "Enter data";

cout << "Enter data";

Input:
4 don't want this
Indiana Jones

x is: " << x
<< "name: "
<< name;
```
cin >> in combo with getline

Code:
```cpp
int x;
string restOfLine;
string name;
cout << "Enter data";

cin >> x;
getline(cin, restOfLine);
getline(cin, name);

cout << "x is: " << x << "name: " << name;
```

Input:
```
4  don't want this Indiana Jones
```

Output:
```
x is: 4
name: Indiana Jones
```
ignore()

Note: this is RARELY USED

- Used if data has "columns"
- Otherwise, DO NOT USE

- Skips (reads and discards) char's in the input stream

- `cin.ignore(howMany, whatChar);`
- skips over `howMany` char's or until `whatChar` is read
- Whichever comes first
You can write a program in C++ to:

- Take input (cin, etc.)
- Compute something interesting
  - Using variables and operators
- Output results (cout, etc.)

Try writing your own program:

- Compute the volumes of various figures
- Compute the cost per area of a pizza
Review

Input/Output
What prints? (Input is on the bottom)
```cpp
cout << "Enter a value: ";
int x = 0;
double z = 5;
char c = ‘ ’;

cin >> x >> z >> c;
cout << "x is: " << x << endl
    << "z is: " << z << endl
    << "c is: " << c << endl;

// 3.14abc
```
What prints? (Input is on the bottom)

```cpp
int x = 0;
double z = 5;
char c = ' ';

cin >> x >> c >> z;
cout << "x is: " << x << endl
     << "z is: " << z << endl
     << "c is: " << c << endl;
// 3.14abc
```
What prints? (Input is on the bottom)
cout << "Enter a value: ";
int x = 0;
double z = 5;
char c = ' ';

cin >> c >> x >> z;
cout << "x is: " << x << endl
    << "z is: " << z << endl
    << "c is: " << c << endl;

// 3.14abc