Today in CS161

- ***Week #3***
  - ***Learn about…***
    - ***Data types (char, int, float)***
    - ***Input and Output (cin, cout)***
  - **Writing our First Program**
    - Write the Inches to MM Program
  - **See example demo programs**

Data Types and Variables

- To work with programs, we need to create variables that will hold information that we will need
- Think of it as memory locations with names
- We can store:

Single character data	char initial;
Whole numbers	int age;
Real numbers	float salary;

Data Types and Variables

- The name in front is called a data type and it represents how much memory to set aside and what can be done with that memory
- *char* will set aside 1 byte of memory and hold 1 character ‘a’, ‘b’, ‘z’, ‘A’, ‘1’, ‘&’ etc.
- *int* will set aside a word of memory and hold a whole number
- *float* will hold something with a decimal component   e.g., 3.14159

Data Types and Variables

- The name after the data type represents the "handle" for how we can access that memory
- So, saying
    - char initial;   //means that I can store a single character
      //and access it through the name "initial"
- The name must start with a letter and be any sequence of letters, digits and underscores:
    - count
    - count_2_numbers
    - my_age

Output Stream

- We can output messages, integers, floating point numbers and characters using the <u>insertion</u> (<<) operator…
- cout << “We did it!”;
- cout << whole_number;
- cout << age;
- cout << salary;
- cout <<endl; //end followed by lower case l

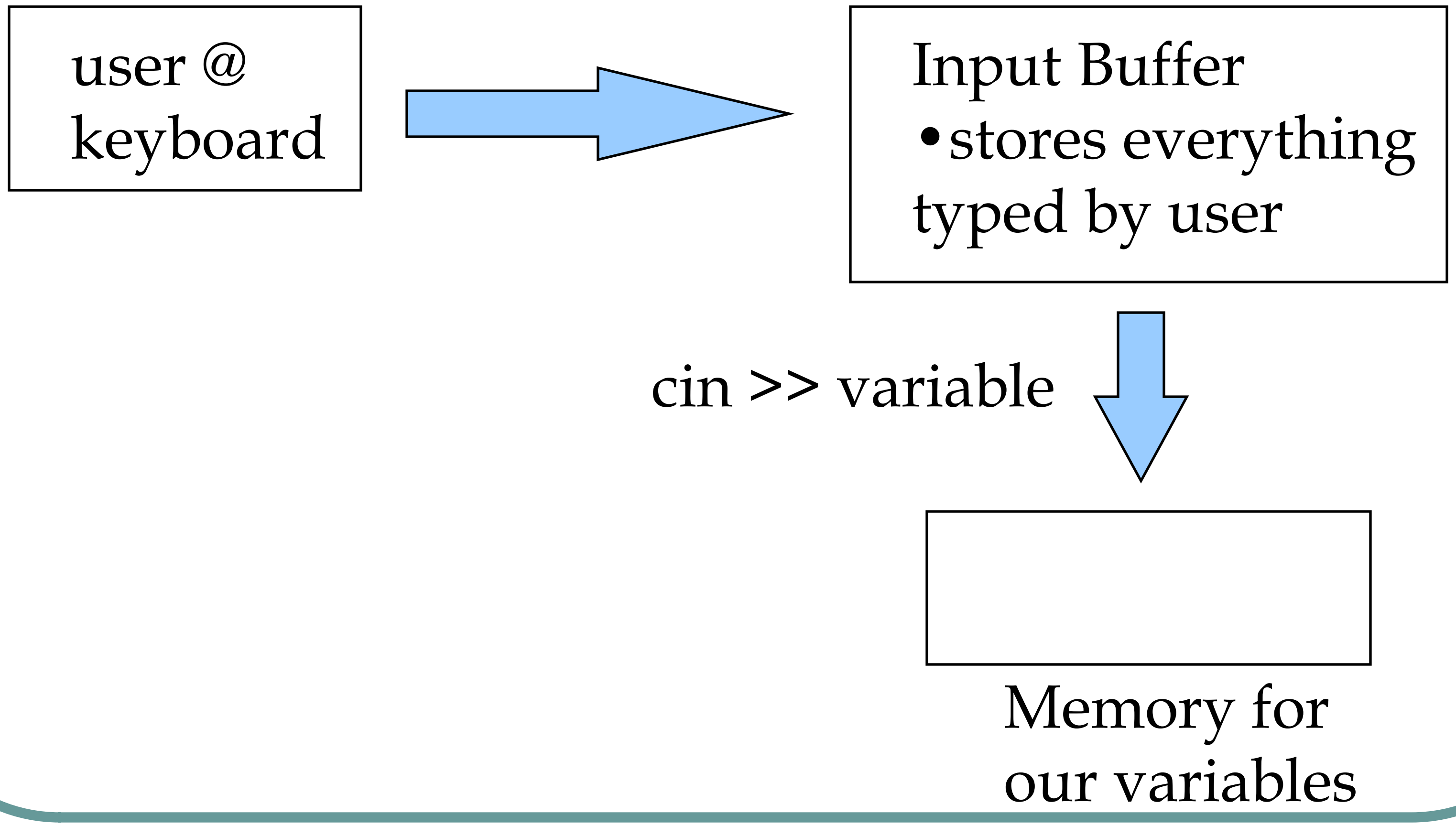
Input Stream

- We can read integers, floating point numbers and characters using the extraction (>>) operator…
- It looks like: cin >> variable;
- We can't, however, control what the user types in.
- Anything the user types in...goes into the input buffer once they hit the enter (or return) key...regardless of what our programs might want!

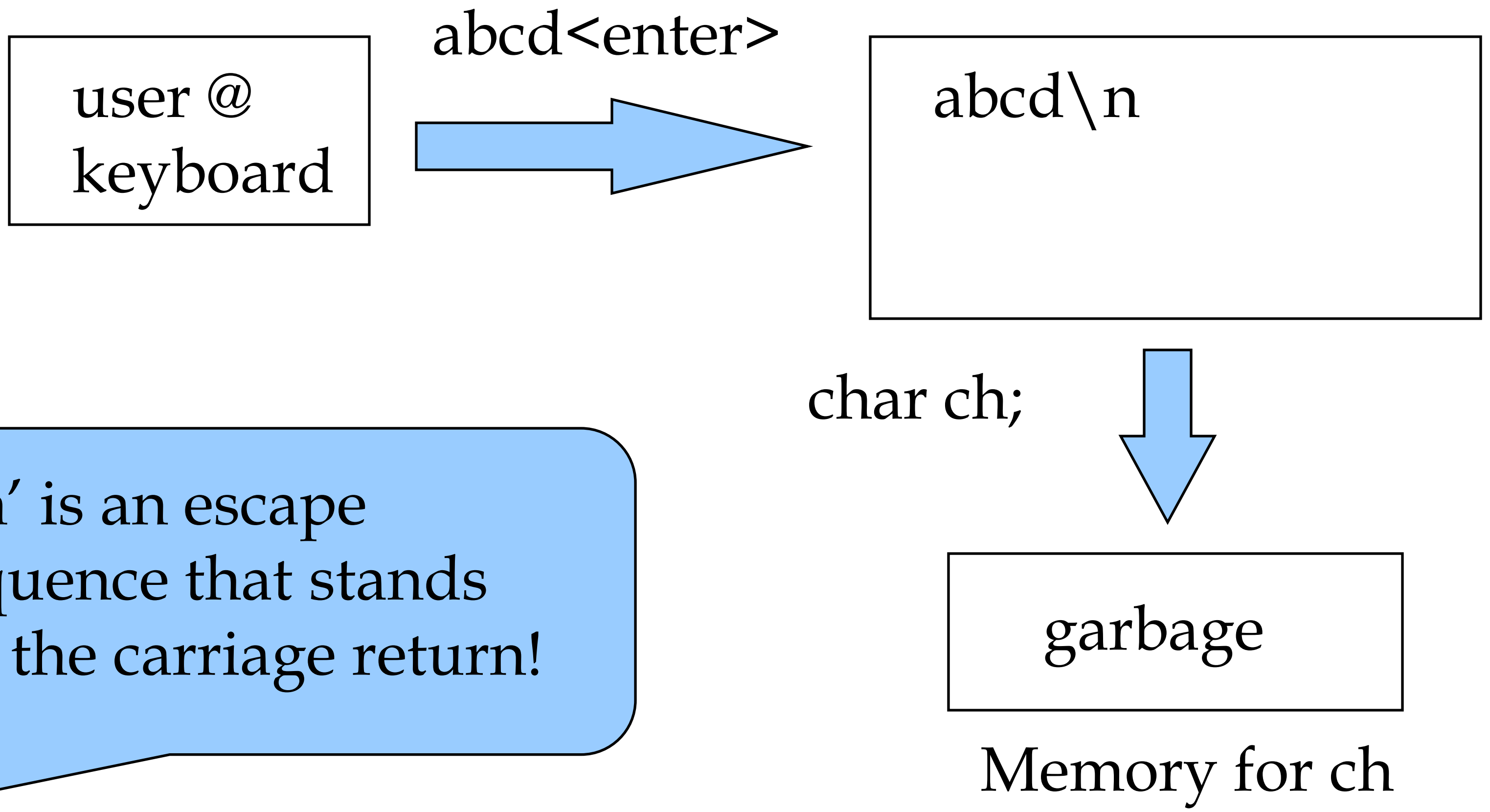
Input Stream

- Therefore, it is important to prompt users, so they know exactly what they are supposed to type in
- And, it is important to understand how input operations behave

Input Stream

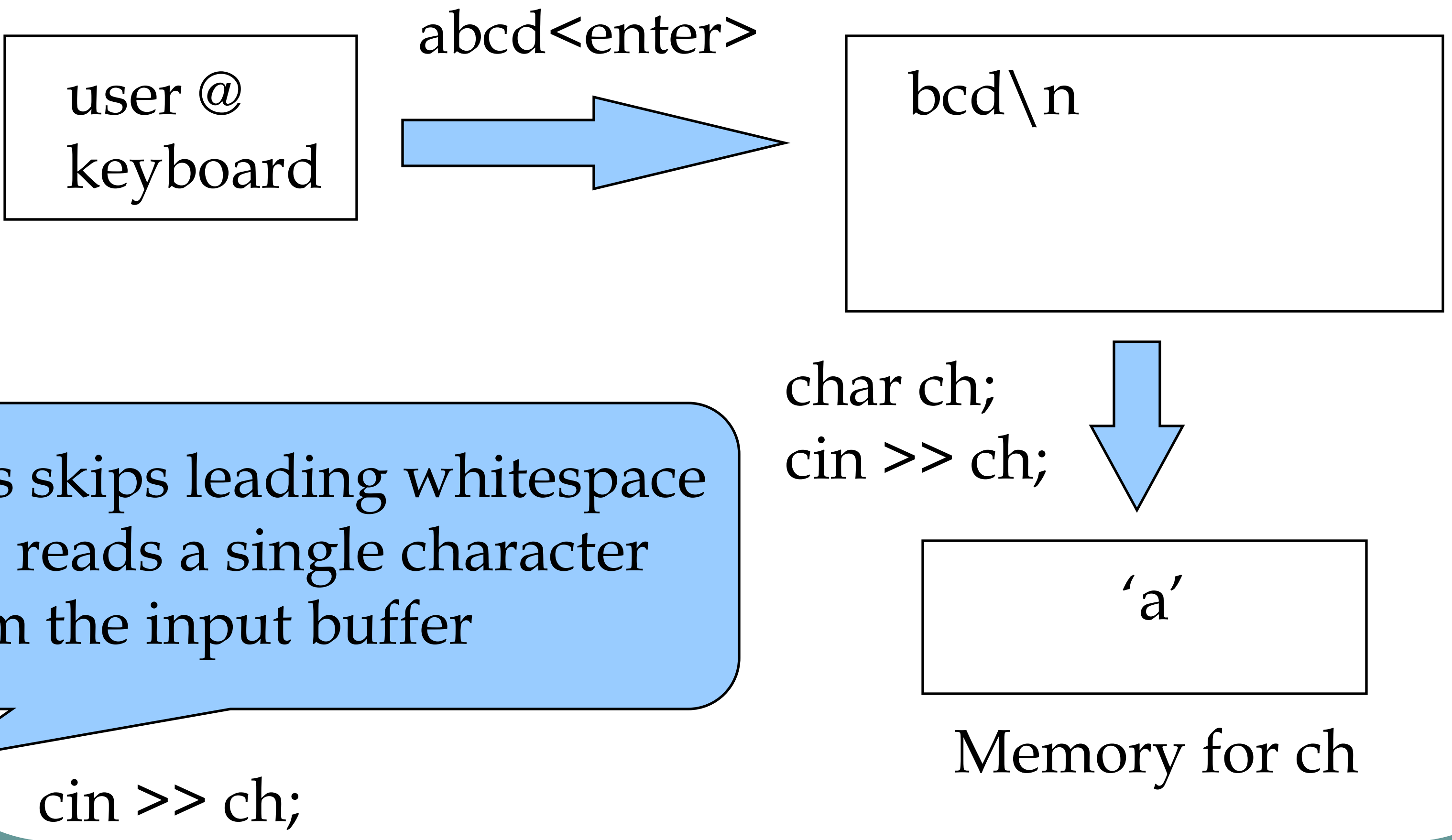


Input Stream



‘\n’ is an escape sequence that stands for the carriage return!

Input Stream


user @ keyboard
abcd<enter>
bcd\n
char ch;
cin >> ch;
'a'
Memory for ch
This skips leading whitespace and reads a single character from the input buffer
cin >> ch;

Input Stream

- What about integers?

```
int  number;
cin >> number;
```

- Skips leading whitespace and reads in digits until it gets to a non-digit, from the input buffer.

Input Stream

- What about floating point numbers?

```
float inches;
cin >> inches;
```

- Skips leading whitespace and reads in digits and optionally one decimal point until it gets to a non-digit or more than one decimal point from the input buffer.

Today in CS161

- ***Applying what we learn to programs***
    - ***Data types (char, int, float)***
    - ***Input and Output (cin, cout)***
- **Writing our First Program**
    - Write the Inches to MM Program
- **See example demo programs**
    - Using graphics!

Now let’s use this in a program!

- Now that we have learned some about
  - Data types
  - Variables
  - Input
  - And, Output

    - *Let’s put it all together!!!!!!!!!!!!!!!*

```
#include <iostream>
using namespace std;

int main()
{
    cout <<"We are going to have a great time!";


    cin.get(); //wait so the window doesn't go away

    return 0;
}
```

```
#include <iostream>
using namespace std;

int main()
{
    int num_classes = 0;  //the number of classes you are taking

    //prompt and read in the number of classes
    cout << “How many classes are you taking this term?”;
    cin >> num_classes;        cin.get();

    //echo what we got back to the user
    cout << “You are taking “ << num_classes << “classes”
            <<endl;

    cout << “Hit ENTER to finish”;

    cin.get(); //wait so the window doesn't go away
```

Convert inches to millimeters

```
#include <iostream>
using namespace std;
//  ***********************************
//   Karla S. Fant
//   CS161 Programming Assignment #0
//   Purpose of this program is to convert
//   inches entered in by the user into
//   millimeters and display the results
//   **********************************
int main() {
```

(Different Kind of Comment...)

```
#include <iostream>
using namespace std;
/*  ***********************************
     Karla S. Fant
     CS161 Programming Assignment #0
     Purpose of this program is to convert
     inches entered in by the user into
     millimeters and display the results
     *********************************** */
int main() {
```

Convert inches to millimeters

```
//Define variables
float inches;          //to save #
  inches
float mm;          //to save the result

//Step #1, welcome the user
```

```
cout <<“Welcome! We will be converting”
    <<“ inches to mm today” <<endl;
```

(A different way to do this…)

```
//Define variables
float inches,                //to save # inches
        mm;                  //to save the result

//Step #1, welcome the user
cout <<“Welcome! We will be converting”;
cout <<“ inches to mm today” <<endl;
```

(NOTE: `endl` is `end` followed by a letter `l`)

Convert inches to millimeters

```
//Step #2, Get the input (prompt, read)
cout <<“Please enter the number of inches”
    <<“ that you wish to convert:  “;

cin >> inches;           //read the # inches
cin.get();               //remove the newline

//echo what was entered
cout <<“You entered: “ <<inches <<“in”
      <<endl;
```

Convert inches to millimeters

```
    //Step #3 Convert inches to millimeters
    mm = 25.4 * inches;

    //Step #4 Display the results
    cout <<inches <<“in converts to “
            <<mm <<“mm” <<endl;

    //Step #5 Sign off message
    cout <<“Thank you for using CONVERT”
            <<endl <<“Hit ENTER to finish!”;
    cin.get(); //wait for user input…
    return 0;
}
```

Next in CS161

- ***Next Topic***
  - ***Learn about…***
    - ***If and else statements***
  - **Rewrite our First Program**
    - Using if and else statements
  - **See example demo programs**

Selective Execution

- **Most programs are not as simple as converting inches to mm!**
- **We need to select from alternatives...**
  - think of the ATM example...
  - this can be done using an **if** statement
  - an **if** allows us to select between 2 choices
  - for example, we can select one thing or another, depending on the user

if Statements

- **For example, we can change our inches to mm conversion program, allowing the user to select whether they want to convert from**
  - inches to mm, or mm to inches!
- **We will give the user a choice...**
  - type ‘m’ to convert to mm
  - type ‘i’ to convert to inches

if Statements have the form...

1) One alternative:

```
if (logical expression)
   single C++ statement;
```

```
char selection;
cout << “Enter a selection (m or i): “;
cin >> selection;
if (selection == ‘i’)    //better to say if (‘i’ == selection)
    cout << “You selected to convert to inches!”
           << endl;
```

if Statements have the form...

2) Two alternatives:

```
if (logical expression)
  single C++ statement;
else
  single C++ statement;
```

```
if (selection == ‘m’)
        cout <<“Converting inches -> mm”;
else
        cout <<“Converting mm -> inches”;
```

if Statements have the form...

- This means that either the first statement is executed when running your program OR the second statement is executed. BOTH sets of statements are NEVER used.
  - ONE OR THE OTHER!

- If the comparison is true - the first set is used;
- If the comparison is false - the second set is used;

if Statements have the form...

- When an if is encountered, the logical expression is TRUE if it is **non zero.** In this case, the statement following the expression is executed.
- Otherwise, if the logical expression evaluates to **zero** it means it is FALSE. In this case, if there is an else the statement following the else is executed.
- If there is no else then nothing is done if the logical expression evaluates to **zero** (FALSE).

if Statements have the form...

3) Two or more alternatives:

```
if (logical expression)
   single C++ statement;
else if (logical expression)
   single C++ statement;
```

```
if (selection == 'm')
        cout <<"Converting inches -> mm";
else if (selection == 'i')
        cout <<"Converting mm -> inches";
```

Compound if statements...

4) You might want more than a single statement to be executed given an alternative...so instead of a single statement, you can use a **compound statement**

```
if (logical expression)
{
  Many C++ statements;
}

else //optional
```

Example of if Statements

```
#include <iostream>
using namespace std;
int main() {
  char selection;  //the user’s answer…one character
  float inches, mm;

  //prompt for input from the user
  cout << “Enter i to convert to inches”
       << “ and m to convert to mm: “;
  cin >> selection; //get the response
  cin.get();
```

Example of if Statements

```
if ('m' == selection) //notice expression!
{
    cout << "Enter the # inches: ";
    cin >> inches;   cin.get();
    mm = 25.4 * inches;  //this is multiplication!
    cout << inches << "in converts to "
        << mm << " millimeters" << endl;

}
```

• • •

Example of if Statements

```
else //selection is not an ‘m’
{
    cout << “Enter the # millimeters: “;
    cin >> mm;  cin.get();
    inches = mm / 25.4;
    cout << mm << “mm converts to ”
         << inches << “ inches” << endl;
}

cin.get(); //wait for user input
```

Or, use the else if sequence…

```
else if (‘i’ == selection) //selection is not an ‘m’
{
    cout << “Enter the # millimeters: “;
    cin >> mm;    cin.get();
    inches = mm / 25.4;   //this is division
    cout << mm << “mm converts to ”
         << inches << “ inches” << endl;
}

else
    cout << “Neither i nor m were selected” << endl;
```

logical expressions

- The comparison operators may be:
  - **Relational Operators:**

    > for greater than

    < for less than

    >= for greater than or equal

    <= for less than or equal
  - **Equality Operators:**

    == for equal to

    != for not equal to

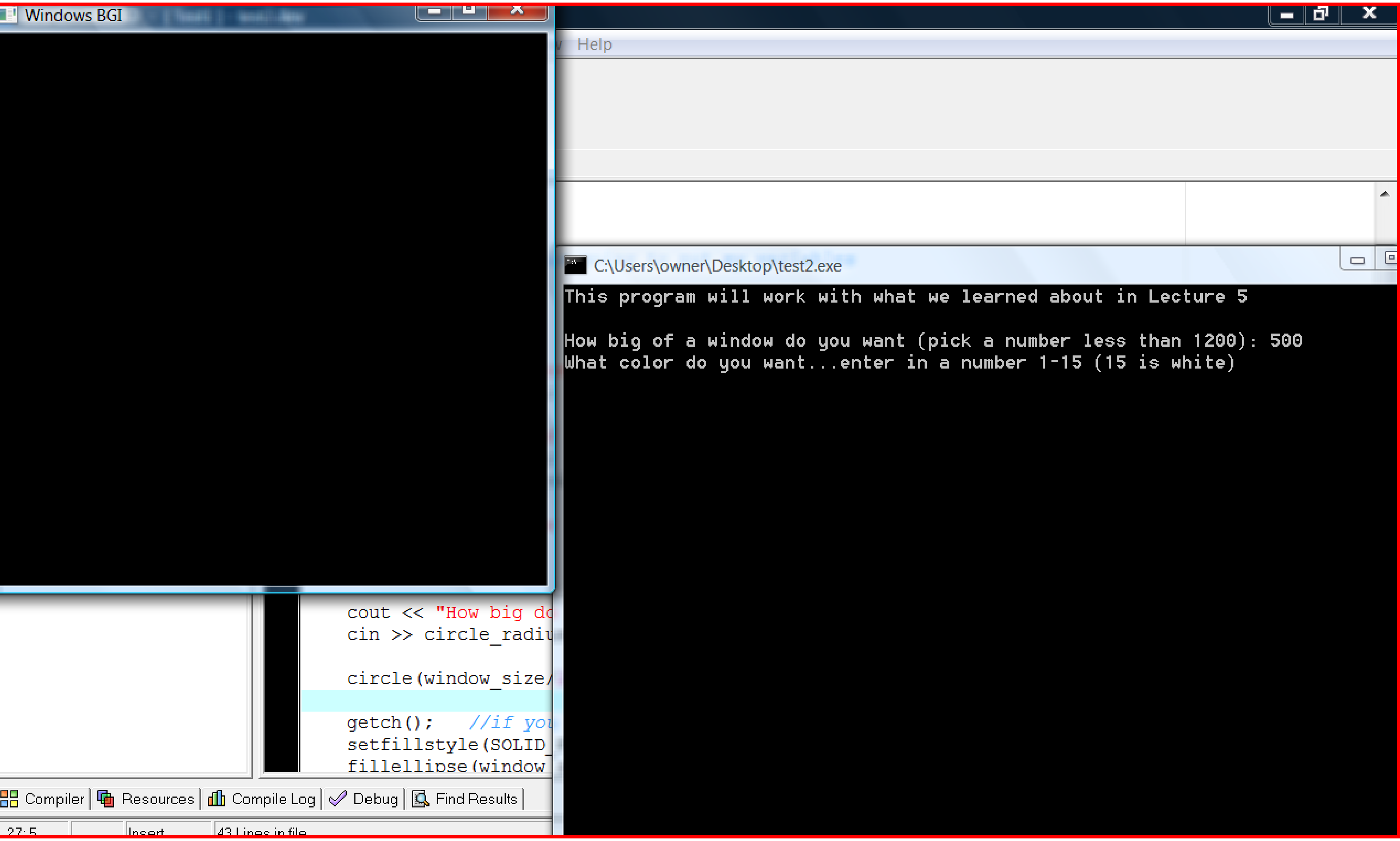
Let’s Write a Graphics Program

- ***New Topic…time for Graphics!***
  - *To create a graphics window you have to use initwindow*
  - *To allow the user to interact and set the size we will use an integer variable:*

```
int window_size;
```

- Then, we will prompt the user for the size:

```
cout << "How big of a window do you want : ";
cin >> window_size;
initwindow(window_size, window_size);
```



```
This program will work with what we learned about in Lecture 5

How big of a window do you want (pick a number less than 1200): 500
What color do you want...enter in a number 1-15 (15 is white)
```

```
cout << "How big do
cin >> circle_radiu

circle(window_size/

getch();    //if you
setfillstyle(SOLID_
fillellipse(window_
```

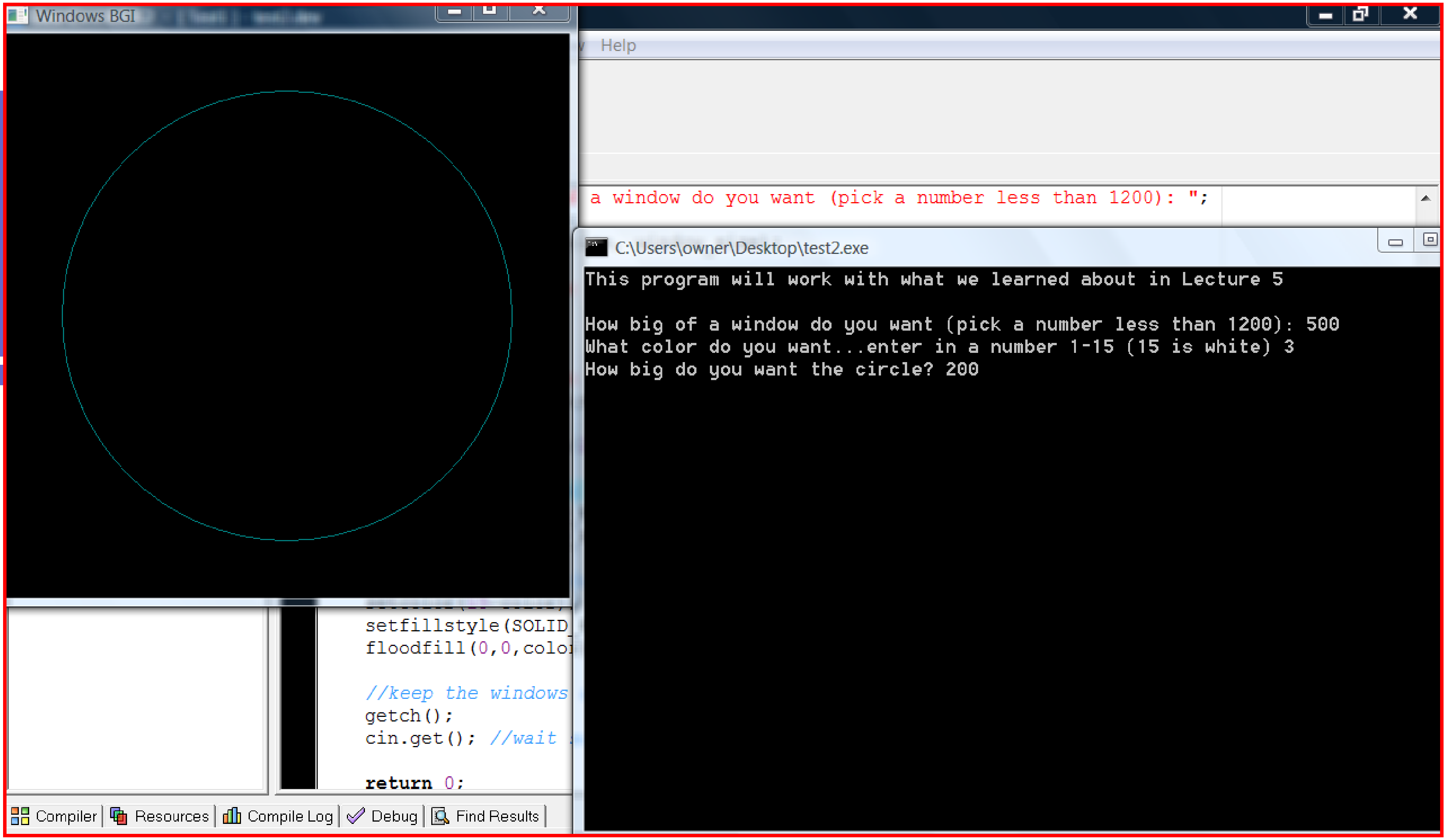
Next, Let’s draw a circle…

- ***Drawing…***
  - *To create an outline of a circle,*
    - *We will ask the user for the radius and then place it at the center of the window (window width divided by 2)*
  - *The circle outline is drawn in the color established by the setcolor function:*

```
cout << "What color do you want...enter in a number 1-15 ";
cin >> color;
setcolor(color);   //the color for the circle

cout << "How big do you want the circle? ";
cin >> circle_radius;

circle(window_size/2, window_size/2, circle_radius);
```



```
a window do you want (pick a number less than 1200): ";
```

```
This program will work with what we learned about in Lecture 5

How big of a window do you want (pick a number less than 1200): 500
What color do you want...enter in a number 1-15 (15 is white) 3
How big do you want the circle? 200
```

```
setfillstyle(SOLID_
floodfill(0,0,colo

//keep the windows
getch();
cin.get(); //wait

return 0;
```

Now Create a Filled Area…

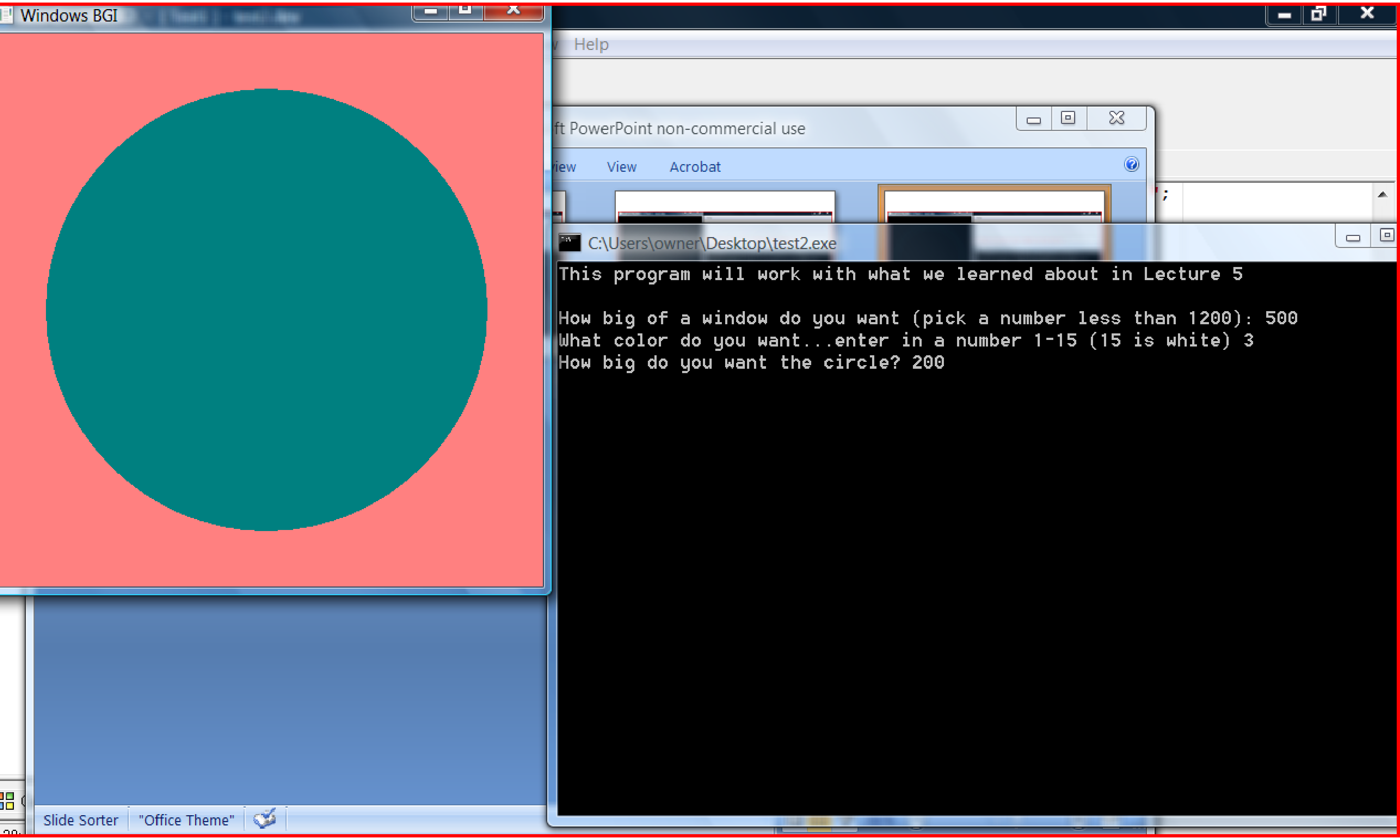
- ***Drawing…***
  - To create filled circle…
    - We can either use the fillellipse function or floodfill
    - Here is an example of both of these
    - They fill in the color set by setfillstyle

```
  setfillstyle(SOLID_FILL,color);

  fillellipse(window_size/2,window_size/2,circle_radius,circle_radius);

//or
  setfillstyle(SOLID_FILL,15-color);  //changing the color….
  floodfill(0,0,color);
```

Hit Enter in the graphics window….


Windows BGI
Help
ft PowerPoint non-commercial use
iew
View
Acrobat
C:\Users\owner\Desktop\test2.exe
Slide Sorter
"Office Theme"

```
This program will work with what we learned about in Lecture 5

How big of a window do you want (pick a number less than 1200): 500
What color do you want...enter in a number 1-15 (15 is white) 3
How big do you want the circle? 200
```

The Complete Program

```
#include "graphics.h"
#include <iostream>
using namespace std;

int main()
{
    //Here is where I am going to put my variables
    int window_size;
    int color;
    int circle_radius;

    cout <<"This program will work with what we learned about in Lecture 5";
    cout <<endl <<endl; //have an extra blank line

    cout << "How big of a window do you want (pick a number less than 1200): ";
    cin >> window_size;
    initwindow(window_size, window_size);
    cout << "What color do you want...enter in a number 1-15 (15 is white) ";
    cin >> color;
    setcolor(color);

    cout << "How big do you want the circle? ";
    cin >> circle_radius;
    circle(window_size/2, window_size/2, circle_radius);

    getch();   //if you don't do this...the first circle disappears!
    setfillstyle(SOLID_FILL,color);
    fillellipse(window_size/2,window_size/2,circle_radius,circle_radius);

    //let's have some fun!
    setfillstyle(SOLID_FILL,15-color);
    floodfill(0,0,color);

    //keep the windows open longer.
    getch();
    cin.get(); //wait so the window doesn't go away
    return 0;
}
```