

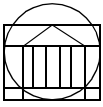
Important Activities

- Read BOCM
- Read Ch. 1 and Ch. 2.1 of Textbook
- Practice Self-test exercises

You do section 1.1 on your own

Topics can/will be on exams.....

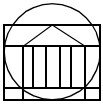
Now on to the material of 1.2



Problem Solving

- Establish the context of the problem
 - understand the problem
 - determine the primary goals
 - create a solution: develop an algorithm
- Algorithm:
A sequence of precise instructions that leads to a solution of a given problem.

- Example: Determine the number of Johns in CS120.
 1. Get the classroll
 2. Set a counter to zero
 3. Do the following for each name in the classroll
 - check whether it is John
 - if it is John, increment the counter
 4. Answer is the number indicated by the counter
- Note that an algorithm is not a program:
Algorithm needs to be translated to C++



Layout of a C++ Program

(Section 1.3)

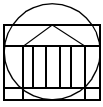
```
// Program File name: layout.cpp
// Author: Your name goes here.
// Lab sect: your lab section #
// Email Address: you@virginia.edu
// Assignment Number: 0
// Description: C++ Program layout
// Last Changed: September 5, 1997

#include <iostream.h>

int main()
    // Program to do something or other.
{
    Variable_Declarations

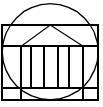
    Statement_1
    Statement_2
    . . . .
    Statement_Last

    return 0;
}
```

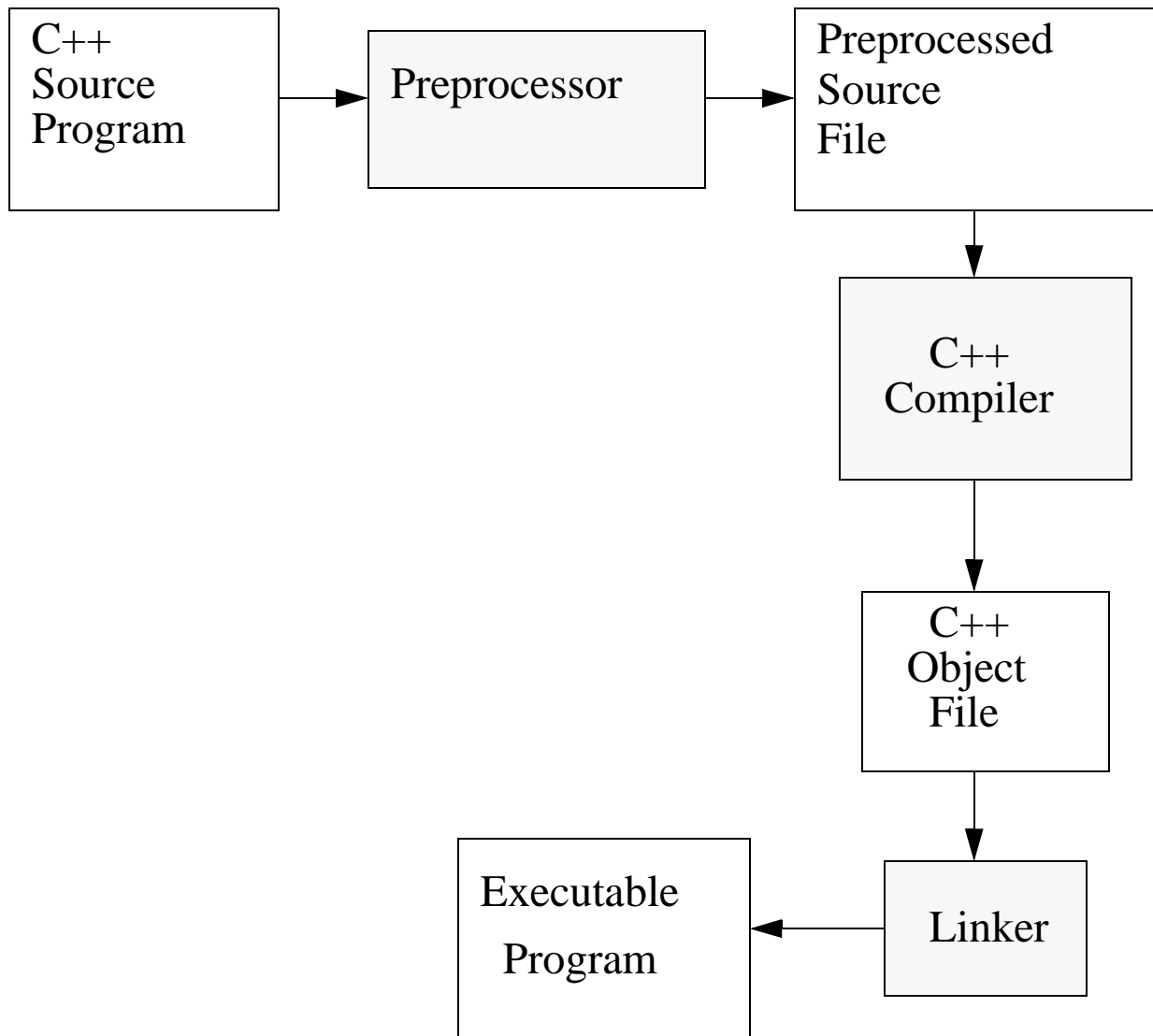


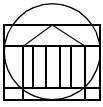
C++

- Program identification
`// Program File name: first.cpp`
identifies the program (usually the file name)
everything following `//` is a comment
- Include standard I/O functions
`#include <iostream.h>`
preprocessor directive
automatically includes chunks of code in your program
- Program header
`int main()` (`int` is a keyword - bolded)
- Program body
`{`
`< body >`
`}`
delimited by `{ ... }`
always indent
always align pairs of `{,}` above each other
- All C++ statements terminated with `;` (semicolon)



Processing a C++ Program





Compiling and Running a C++ Program

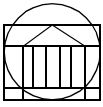
- Enter a C++ program using editor (*source* program)
- Save it in a file
File name: a:\cs120\first.cpp
- Compile a source program and link it
object code
executable program
- Run
execute the program

Now.... if things don't go right...(they rarely do!!)

- Syntax errors
program violates C++ syntax rules
compiler gives error or warning messages
- Run-time errors
can be detected only when a program is executing

Logic errors

mistakes in algorithm or translation into C++
program does not do what it supposed to do

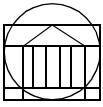


C++ Program

```
// File name: first.cpp
// Author: Yo Name.
// Lab sect: 9
// Email Address: ykn7b@virginia.edu
// Assignment Number: 1
// Description: Program to make you feel good.
// Last Changed: September 5, 1997

#include <iostream.h>

int main()
    // Program to print two lines.
{
    cout << "Hello Programmer!\n";
    cout << "You are brilliant!\n";
    return 0;
}
```



Input and Output in C++

- C++ input statement

```
cin >> number;
```

a numerical value is *extracted* from the keyboard (cin) and is placed into the variable called "number".

- C++ output statement

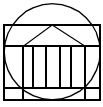
```
cout << "Hello World!\n";
```

send information from program to terminal screen
or: characters from the program are *inserted* into
the monitor object (cout)

double quotes " ... " delimit a string
the '\n' is an end-of-line character

- Arrows << and >> show the direction of data movement
- cin and cout are predefined in `iostream.h`

We will discuss more about input and output later.



Variables

Containers for information

Used for "remembering" things during program execution

Attributes

- Name (name of the container.)
- Type (what kinds of things can it contain?)
- Value (what does it contain?)

Two different variables can have:

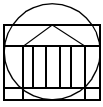
- same type
- same value

Variable's value can change during program execution

- hence "variable"

name
(address)

a value is in here.....



Second C++ Program

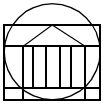
```
// File name: second.cpp
// Author: Yo Name.Lab sect: 9
// Email Address: ykn7b@virginia.edu
// Assignment Number: 2
// Description: Program to display a
//             number you typed.
//Last Changed: January 22, 1886

#include <iostream.h>

int main()

//Program to echo the input value.

{
    int number_typed;
    cout << "Hello Programmer!\n";
    cout << "Enter number>";
    cin >> number_typed;
    cout << endl;
    cout << "The number you typed was "
           << number_typed << endl;
    return 0;
}
```



Literals

- A value used in a program
- Three basic types in C++: numeric, character and string
- *Numeric* is a number

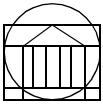
```
5  
3.14159  
-1729
```

- *Character* is a single symbol

```
'a'  
'7'  
'*'
```

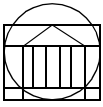
- *String* is a sequence of symbols

```
"I will be a C++ expert"
```



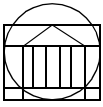
Literals and Variables Compared

- A literal *is* a value
 - the value never changes
- A variable is a container for values
 - a variable is *named*
 - value of a variable *can* change
 - assignment statement is one way
 - there are many others
- A variable can only hold one value at a time
- Variable loses its old value when given new one
- Variables **must** be declared



Identifiers

- Identifiers are used as names for variables and other items
 - must start with either a letter or underscore symbol
 - x, x_1, _Student, rate, BIG_NUMBER --- **legal**
 - 2BORN0T2B, %new, My.program, data-1 --- **illegal**
 - \$foo, TooHotToHandle? --- **illegal**
- Conventions
 - names should be descriptive
 - use "_" as a separator
 - Try not to reuse names
- C++ is case-sensitive
 - rate RATE Rate
 - three distinct variable identifiers; confusing
- C++ identifiers can be of any length
 - try to make it short but meaningful
 - my_best_score_of_CS120_Fall_1997 -- too much!

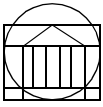


Reserved Words (Keywords)

- Reserved - don't use for other purposes
- Always in lower case
- Common reserved words:
(complete list in Appendix 1, page 769)

break	case	char
const	default	do
double	else	extern
float	for	if
int	long	return
switch	void	while

We will come back to these later.



Variable Declarations

- Every variable in C++ must be declared

normally occurs at the start of a main program
specifying the name of the object and what kind values
the object will be allowed to contain

- Syntax:

```
type_name    variable_name;
```

type_name is a basic type: char, int,
double, ...

variable_name is the name of the variable

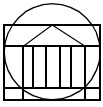
```
int number_of_cats, count;
```

```
double distance;
```

For the moment, we will consider only the types:

int (the integers, or counting things)

double (all the real numbers)



Simple Assignment Statement

Common form

`variable = expression;`

- Causes

expression to be evaluated and the result assigned as the new value of the *variable*

- Examples

```
i = 5;           // var i's value set to 5
i = i + 1;      // increment i by 1
y = m*x + b;    // just as it appears!
```

```
// Swap x and y
x = y; // WRONG!!
y = x;
```

```
temp = x; // RIGHT
x = y;
y = temp;
```

Can't Do's:

```
5 = 7;
5 = x;
"I will get this" = "Not now";
```