Introduction to C++

A Basic Program

01/15/2009

Kill Leftover VNC Sessions

Connect to `comphy.fsu.edu:#`:
("#" is your vnc port number)

- Check your vncserver(s)
  - If a vncserver is up and running there should be a file called `~/.vnc/comphy.fsu.edu:#.pid`
- On comphy, run the command

  "ps -aux | grep vnc"

  and look for your user name

  (or type "ps aux | grep vnc | grep <username>")

Outline

Homework Assignment

1. Hopefully, you have read Chapter 3 & 4 !!
   - *Introduction to Computer and Software Architecture*
   - *Fundamental C++ Concepts*
2. Assignments of Chapters 3 & 4
   - See handout!
   - Due Tuesday, January 20
     ➜ Hand in a paper copy

Outline

Tokens, Names, and Keywords

```
# include <iostream.h>

main() {
    int var = 4;

    var = var + 1;
    cout << endl << var;
}
```

1. A program is read as a sequence of characters called tokens: a-z, A-Z, 0-9, and punctuation characters.
2. A word or name is a sequence of tokens terminated by *whitespace*.
3. A reserved keyword is a word, such as **int** or **cout**, for which the compiler has a special interpretation.

Expressions and Statements

```
# include <iostream.h>

main() {
    int var = 4;

    var = var + 1;
    cout << endl << var;
}
```

1. A segment of code in C++ that is terminated by a semicolon is called a statement.
2. Preprocessor directives begin with a # sign and are not terminated by a semicolon.
3. A statement is composed of one or more valid expressions separated by operators ($+$, $-$, $*$, $/$, $<<$):
   - 13; is valid statement
   - a = b + 3; "expression"

Declarations and Definitions

A declaration tells the compiler about the properties of a name. A definition, in addition, specifies the value to be associated with a name, or in the case of a variable definition, specifies that storage should be allocated.

```
# include <iostream.h>

main() {
    int var = 4;    This is a definition and includes a declaration.

    var = var + 1;
    cout << endl << var;
}
```

Declarations and Definitions

A declaration tells the compiler about the properties of a name. A definition, in addition, specifies the value to be associated with a name, or in the case of a variable definition, specifies that storage should be allocated.

```
# include <iostream.h>

main() {
    int var = 4;
    int var = 4;    Compiler error, "var" can only be defined once!

    var = var + 1;
    cout << endl << var;
}
```

Declarations and Definitions

A declaration tells the compiler about the properties of a name. A definition, in addition, specifies the value to be associated with a name, or in the case of a variable definition, specifies that storage should be allocated.

```
# include <iostream.h>

extern int var;    But var can be declared more often!

program() {

}
```

Declarations and Definitions

A declaration tells the compiler about the properties of a name. A definition, in addition, specifies the value to be associated with a name, or in the case of a variable definition, specifies that storage should be allocated.

```
# include <iostream.h>

main() {
    int var = 4;
    var = 10;    Not a definition, does not allocate new memory.

    var = var + 1;
    cout << endl << var;
}
```

rvalues and lvalues

The term *rvalue* refers to a data value that is stored at some address in memory. An *rvalue* is an expression that cannot have a value assigned to it. Both a literal constant and a variable can serve as an *rvalue.* When an *lvalue* appears in a context that requires an *rvalue,* the *lvalue* is implicitly converted to an *rvalue.*

```
# include <iostream.h>

main() {
    int var = 4;

    var = var + 1;
    cout << endl << var;
}
```

Outline

Formatting Conventions and Comments

Comments: /* ... */ or //

```
// What does this program do?
// Author:
// Creation Date:
# include <iostream.h>

main() {
    int var = 4;

    var = var + 1;
    cout << endl << var;
}
```

1. Prolog
2. Section explanation
3. Interpretation of significant code

Formatting Conventions and Comments

Comments: /* ... */ or //

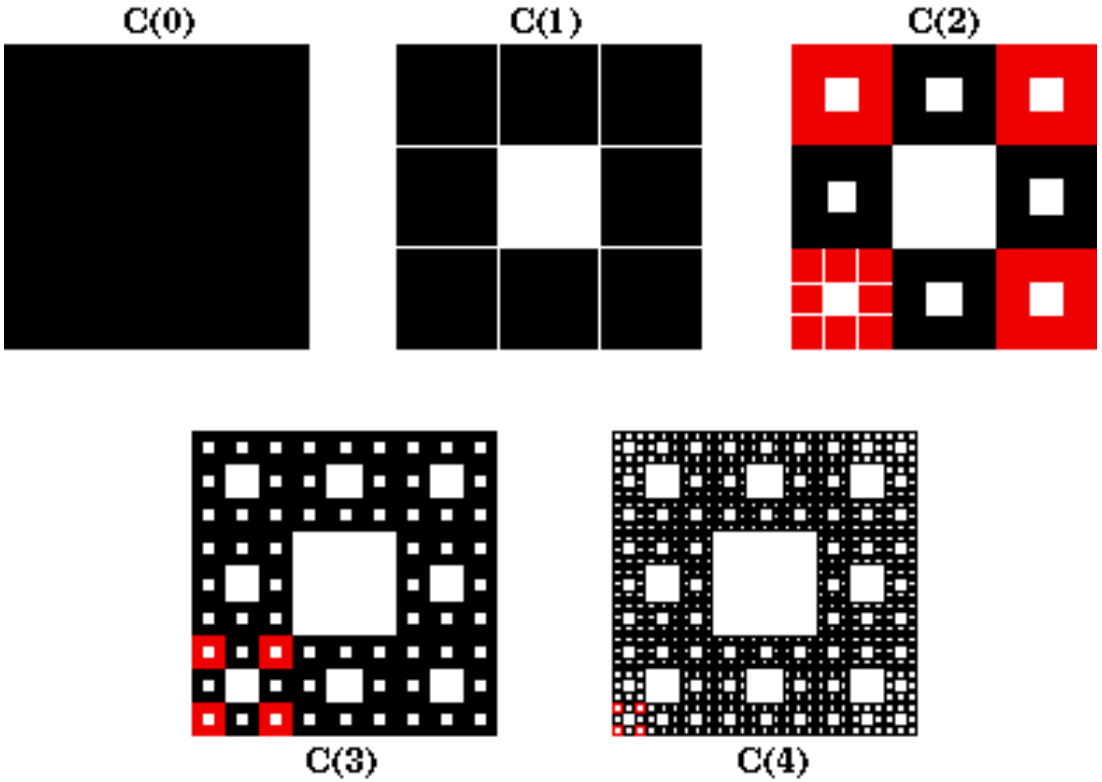
4.10 Conventions ("The 10 Formatting Commandments"):

- Descriptive variable or function names: numberOfGridPoints
- Spaces to the right and left of binary operators: i = j + −10
- ...
- Blank lines should be used between different program units.
- Function arguments should begin with a small a.
- ...
- All letters of global constants should be capitalized.

Outline

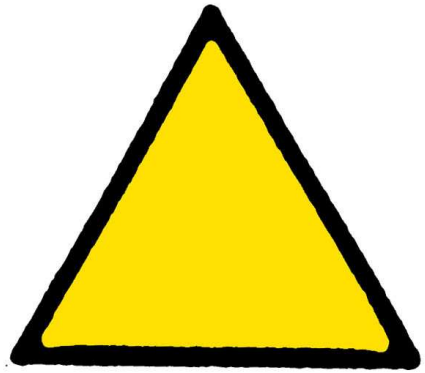
Programming Standards & Simple Data Plots

Sierpinski's Carpet



Programming Standards & Simple Data Plots

Sierpinski's Carpet



1. Select a starting point in the triangle
2. Randomly choose a vertex
3. Make new point halfway to vertex
4. Repeat from "2"

Program Building & Running

Programming and Compiling

```
[<user>@comphy ~]$ emacs sierpinski.cc &
[<user>@comphy ~]$ g++ -Wall -pedantic sierpinski.cc -o sierpinski
```

Executing Program

```
[<user>@comphy ~]$ sierpinski
0.5 0
0.25 0
0.125 0
...
[<user>@comphy ~]$ sierpinski > sierpinski.dat
```

Plotting with Gnuplot

```
[<user>@comphy ~]$ gnuplot
gnuplot> plot "sierpin.dat" with points ps 0.1
```

Saving graph output in gnuplot

```
gnuplot> set output "sierpin.pbm"
gnuplot> set terminal pbm color
gnuplot> plot "sierpin.dat" with points ps 0.1
gnuplot> set terminal x11
gnuplot> quit
```