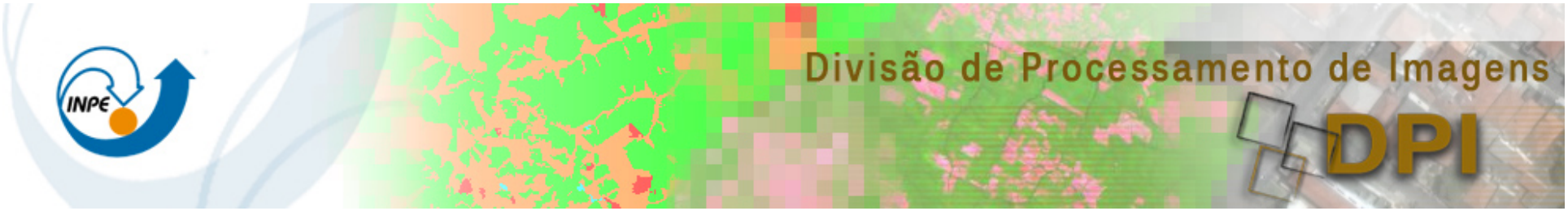
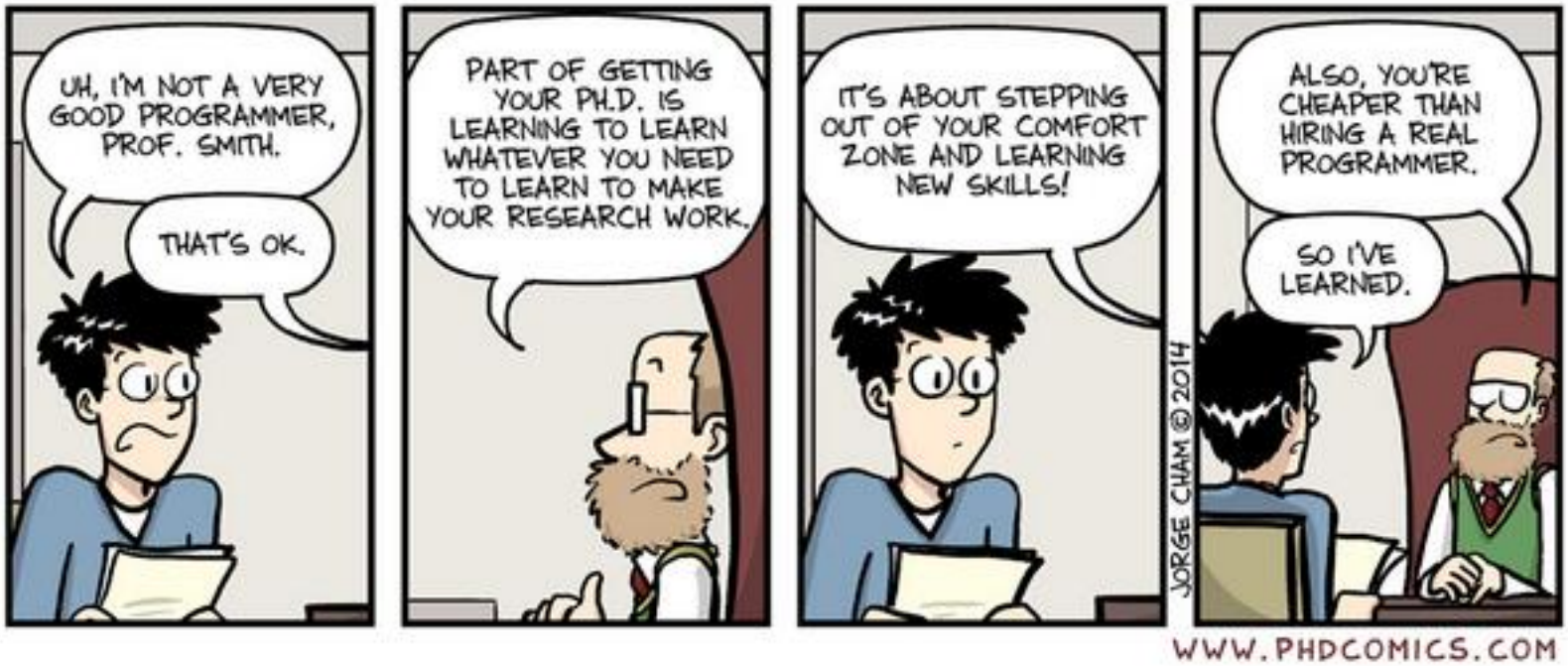


CAP- 390-1 Fundamentos de Programação Estruturada

Lubia Vinhas


UH, I'M NOT A VERY GOOD PROGRAMMER, PROF. SMITH.
THAT'S OK.
PART OF GETTING YOUR PH.D. IS LEARNING TO LEARN WHATEVER YOU NEED TO LEARN TO MAKE YOUR RESEARCH WORK.
IT'S ABOUT STEPPING OUT OF YOUR COMFORT ZONE AND LEARNING NEW SKILLS!
ALSO, YOU'RE CHEAPER THAN HIRING A REAL PROGRAMMER.
SO I'VE LEARNED.
JORGE CHAM © 2014
WWW.PHDCOMICS.COM

Resources

- References:
  - Ascencio, A. F. G., Veneruchi, E. A. Fundamentos de Programação de Computadores. São Paulo: Prentice-Hall, 3a. Edição, 2012.
  - Stroustroup, B. Programming: Principles and Practice Using C++. 2nd Edition. Addison-Wesley. 2014.
  - Stroustroup, B. The C++ Programming Language (4th Edition). Addison-Wesley Professional. 2013.
- Wiki: *http://wiki.dpi.inpe.br/doku.php?id=cap386*
- Classes on Wednesdays from 08:15 to <= 10:00
- Send me an email from an address that you see: lubia@dpi.inpe.br

This is a course

- In Programming
- For beginners
  - who want to become professionals
    - i.e., people who can produce systems that others will use
  - who are assumed to be bright
  - who are willing to work hard
- Using the C++ programming language
- It is NOT:
  - a course in C++ programming language
  - for students who want o become language lawyers
  - using ome untested software development

The Aims

- Teach/learn
  - Fundamental programming concepts
  - Key useful techniques
  - Basic Standard C++ facilities
- After the course, you'll be able to
  - Write small colloquial C++ programs
  - Read much larger programs
  - Learn the basics of many other languages by yourself
  - Proceed with an "advanced" C++ programming course
- After the course, you will not (yet) be
  - An expert programmer
  - A C++ language expert
  - An expert user of advanced libraries

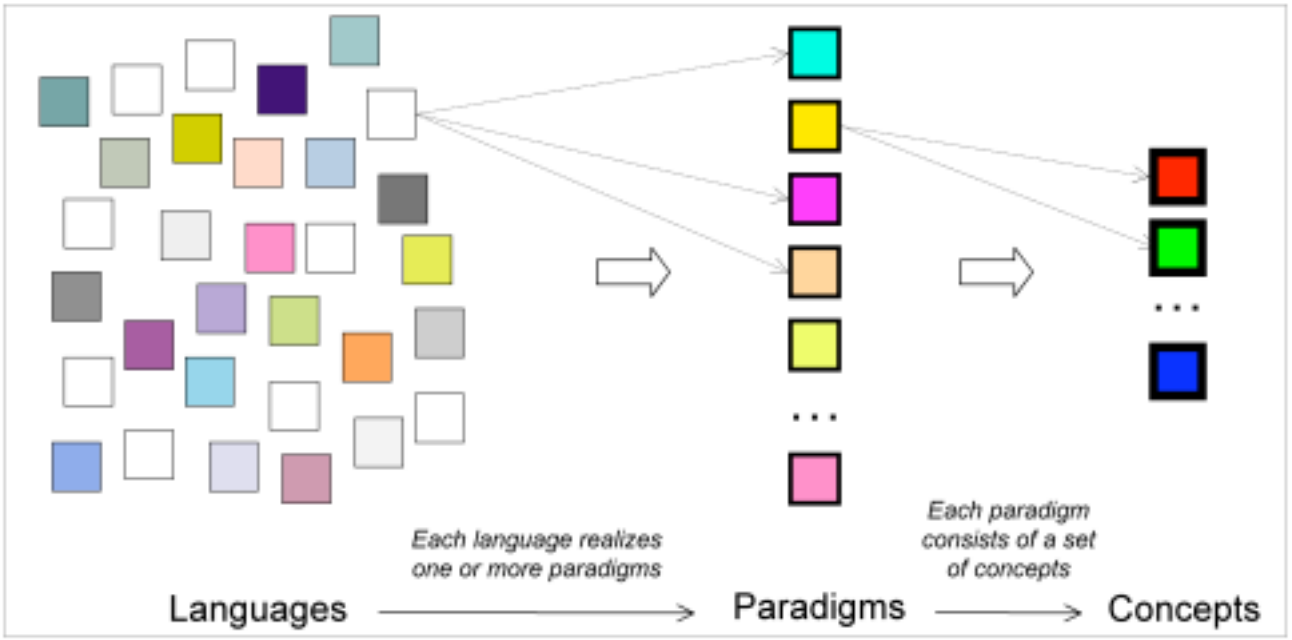
Some terms

- **Programming paradigm**: a pattern that serves as a school of thoughts for programming of computers
- **Programming technique**: related to an algorithmic idea for solving a particular class of problems
  - Examples: 'Divide and conquer' and 'program development by stepwise refinement'
- **Programming style**: the way we express ourselves in a computer program
  - Related to elegance or lack of elegance
- **Programming culture**:
  - The totality of programming behavior, which often is tightly related to a family of programming languages
  - The sum of a main paradigm, programming styles, and certain programming techniques

Programming Paradigms

- Solving a programming problem requires choosing the right concepts
- All but the smallest toy problems require different sets of concepts for different parts
- A programming paradigm, or programming model, is an approach to programming a computer based on a mathematical theory or a coherent set of principles
- Languages realize programming paradigms
- There are many fewer programming paradigms that programming languages

Languages, Paradigms and Concepts



source: Roy, P.V; Haridi, S.; 2004

Programming Paradigm

- **Imperative**: tell the computer how to do obtain an output
- **Declarative**: tell the computer the result that you need

- A way of conceptualizing what it means to perform computation and how tasks to be carried out on the computer should be structured and organized
- Four main paradigms:
  - Imperative : Machine-model based
  - Functional : Equations; Expression Evaluation
  - Logical : First-order Logic Deduction
  - Object-Oriented : Programming with Data Types

Imperative paradigm

First do this and next do that

- Based on the ideas of a Von Neummann architecture
- A command has a measurable effect on the program
- The order of commands is important
- Characteristics:
  - Incremental change of the program state (variables) as a function of time
  - Execution of commands in an order governed by control structures
  - Procedures: abstractions of one or more actions, which can be called as a single command
- Representatives: Fortran, Algol, Basic, C, Pascal

Example


PROGRAMMING FOR NON-PROGRAMMERS
INCLUDE: LIMITED_PROGRAMMING SKILLS
when does it need to get done?
DEFINE: DEADLINE
NOW would be good.
LOAD CODE BY PREVIOUS PERSON ON PROJECT
huge mess
WHILE Project not(FINISHED) input = rand(typing)
SET self_esteem: LOW
I'm a hack!
FOR (time = ∞) DEBUG DEBUG DEBUG
IF code(works) ->EUPHORIA++
woooo!
hello world
SET self_esteem: ABOVE NORMAL
I am... THE ONE.
IF (NOT) organize + comment code for next person on project
yeah right.
JORGE CHAM © 2014
WWW.PHDCOMICS.COM

Example in Pascal Language

```
1   Program Example1;
2   Var
3       Num1, Num2, Sum : Integer;
4
5   Begin {no semicolon}
6    Write('Input number 1:');
7    Readln(Num1);
8    Writeln('Input number 2:');
9    Readln(Num2);
10   Sum := Num1 + Num2; {addition}
11   Writeln(Sum);
12   Readln;
13  End.
```

Most popular programming languages are imperative.

Functional paradigm

Evaluate an expression and use the resulting value for something

- Mathematics and theory of functions
- The values produced are non-mutable
- Time plays a minor role compared to imperative program
- All computations are done by applying functions with no side effects
- Functions are firsts class citizens
- Representatives: Haskell, Clojure

Example in Haskell language

JavaScript

```javascript
function evenSum(list) {
    var result = 0;
    for (var i=0; i< list.length ; i++) {
        if (list[i] % 2 ==0) {
            result += list[i];
        }
    }
    return result;
}
```

Haskell

```haskell
-- Version 4
evenSum :: Integral a => [a] -> a

evenSum = accumSum 0
    where
        accumSum n [] = n
        accumSum n (x:xs) =
            if even x
               then accumSum (n+x) xs
               else accumSum n xs
```

Logical paradigm

Answer a question via search for a solution.

- The logic paradigm fits well when applied in problem domains that deal with the extraction of knowledge from basic facts and relations.
- Based on axioms, inference rules, and queries.
- Program execution becomes a systematic search in a set of facts, making use of a set of inference rules
- Representatives: Prolog

Example in Prolog

```
mother(alice,lea).
mother(john,julia).
mother(lea,alberta).
father(james,alfred).
father(lea,john).
```

```
parent(X, Y) :- father(X, Y).
parent(X, Y) :- mother(X, Y).
```

```
grandparent(X, Y) :- parent(X, Z), parent(Z, Y).
```

```
?- grandparent(X, Y).
X=alice,
Y=alberta ; /* alberta is alice's grandma */
X=alice,
Y=john. /* john is alice's grandpa */
```

Object-Oriented paradigm

Send messages between objects to simulate the temporal evolution of a set of real world phenomena

- Data as well as operations are encapsulated in objects
- Information hiding is used to protect internal properties of an object
- Objects interact by means of message passing
- In most object-oriented languages objects are grouped in classes
- Classes are organized in inheritance hierarchies
- Representatives: C++, Java

Example in C++

```
class Employee
{
  public:
    Employee();
    const char *getName() const;
  private:
    char name[30];
};

class WageEmployee : public Employee
{
  public:
    WageEmployee(const char *nm);
    void setWage(double wg);
    void setHours(double hrs);
  private:
    double:wage;
   double:hours;
};

WageEmployee aWorker("Bill Gates");
const char *str;

aWorker.setHours(40.0);  //call WageEmployee::setHours
str = aWorker.getName(); //call Employee::getName
```

Other paradigms

- The visual paradigm
- One of the parallel paradigms
- The constraint based paradigm
- Aspect-oriented paradigm
- Event-oriented paradigm
- …

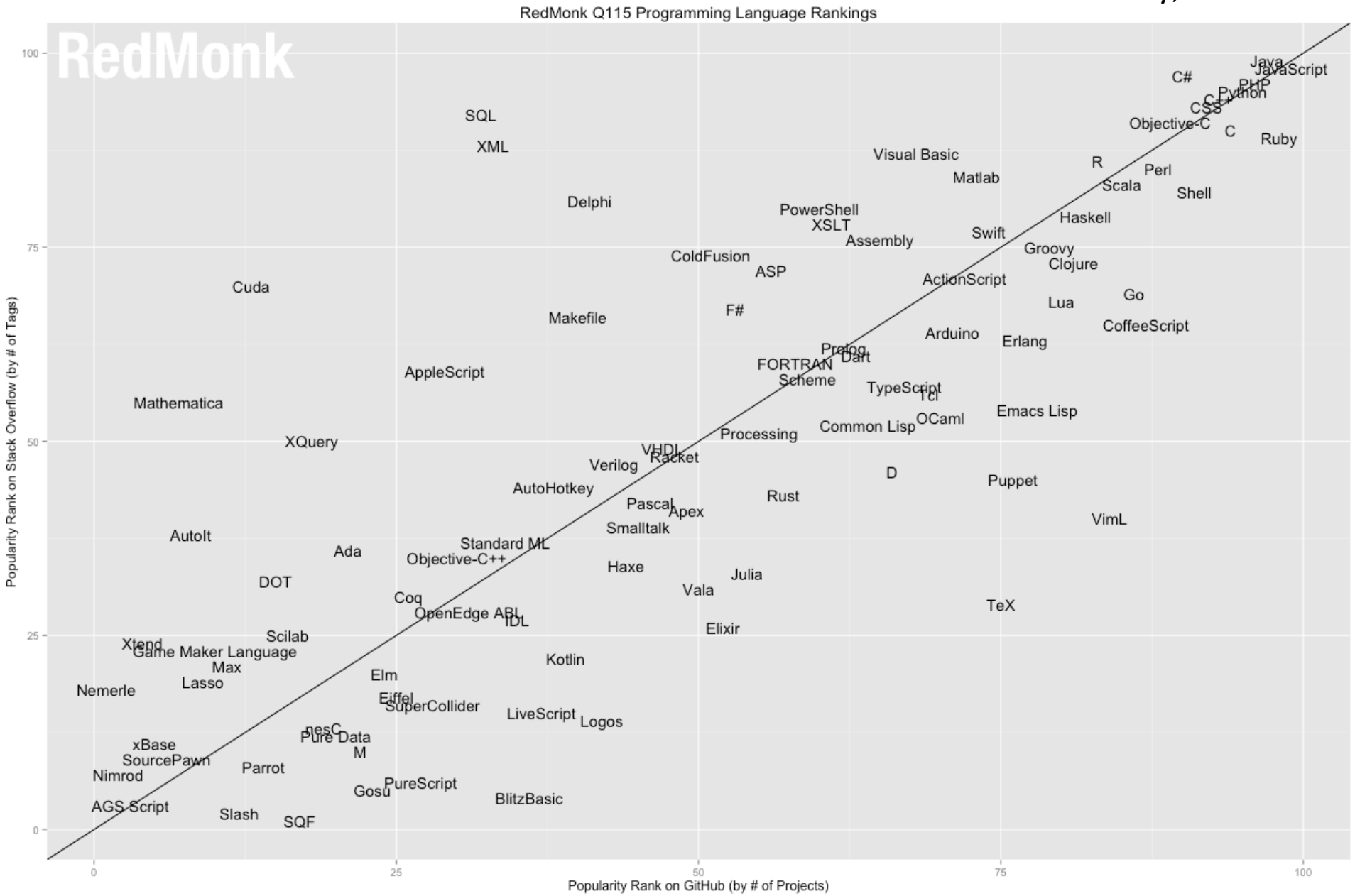
Programming Languages

- What we have to learn to study a programming language?
  - Syntax…
  - Semantics…
  - The programming environment…
- How many languages are out there?

C	PHP
C++	Haskell
Java	Prolog
Python	???

- Which languages should I know?

January, 2015



source: http://sogrady-media.redmonk.com/sogrady/files/2015/01/lang.rank_.plot_.q1152.png

The top 20

source: http://redmonk.com/sogrady/category/programming-languages/

1) JavaScript	6) C	11) Perl	16) Matlab
2) Java	7) Ruby	12) Shell	17) Clojure
3) Python	8) CSS	12) Scala	18) Go
4) C#	9) Objective C	14) Haskell	19) Visual Basic
5) C++	10) PHP	15) R	20) Groovy

Visit http://langpop.com/

- Any ranking is influenced by:
  - Communities of the development
  - Investments from third parties and ubiquitousness of projects
  - Statistics

The top 20

source: http://redmonk.com/sogrady/category/programming-languages/

1) JavaScript	6) C++	11) Perl	16) Matlab
2) Java	7) Ruby	12) Shell	17) Clojure

This course is not about advocating the use of this or that language...

I first learnt how to program with algorithms, than I learnt PASCAL
I work with C++, SQL and I can do a few things in PHP
I believe I can learn other languages if I have to… ☺

I am not a radical champion for C++ or any other language...
I still believe in "*No Silver Bullet — Essence and Accidents of Software Engineering*"
This course will teach C++.

V

- investments from third parties and of projects ubiquitousness
- Statistics

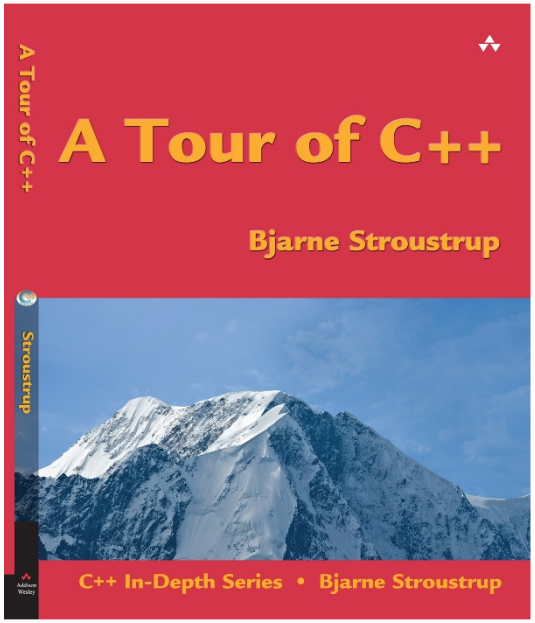
Why C++ ?

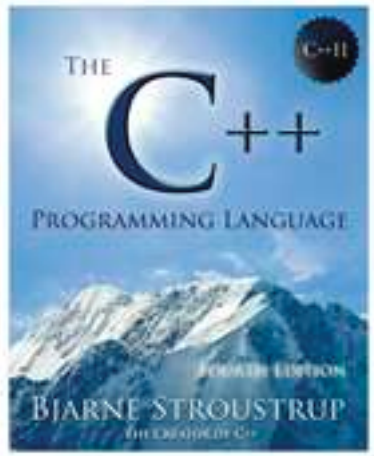
- You can't learn to program without a programming language
- The purpose of a programming language is to allow you to express your ideas in code
- C++ is the language that most directly allows you to express ideas from the largest number of application areas
- C++ is available on almost all kinds of computers
- Programming concepts that you learn using C++ can be used fairly directly in other languages
  - Including C, Java, C#, and (less directly) Fortran

Introducing The Creator

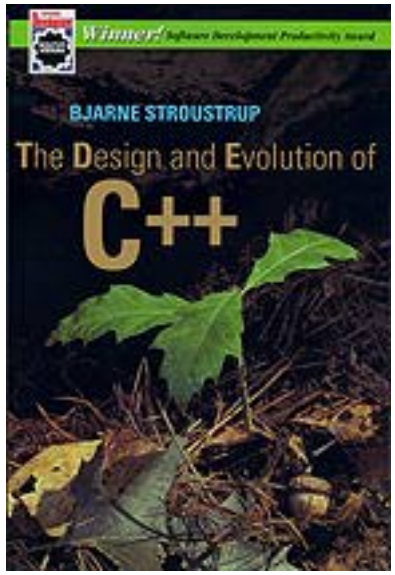
- Bjarne Stroustrup is the creator of C++
- http://www.stroustrup.com/
- http://www.cplusplus.com/info/history/

Books





*



* I love this book! BS had make teaching materials available. I am using it.

The C++ Language

- ...is an open ISO-standardized language
  - for a time, C++ had no official standard and was maintained by a de-facto standard, however since 1998, C++ is standardized by a committee of the ISO.
- ...is a compiled language.
  - C++ compiles directly to a machine's native code, allowing it to be one of the fastest languages in the world, if optimized.
- ...is a strongly-typed unsafe language.
  - C++ is a language that expects the programmer to know what he or she is doing, but allows for incredible amounts of control as a result.

The C++ language

- ...supports both manifest and inferred typing.
  - As of the latest C++ standard, C++ supports both manifest and inferred typing, allowing flexibility and a means of avoiding verbosity where desired.
- ...supports both static and dynamic type checking.
  - C++ allows type conversions to be checked either at compile-time or at run-time, again offering another degree of flexibility. Most C++ type checking is, however, static.
- ...offers many paradigm choices.
  - C++ offers remarkable support for procedural, generic, and object-oriented programming paradigms, with many other paradigms being possible as well.

The C++ Language

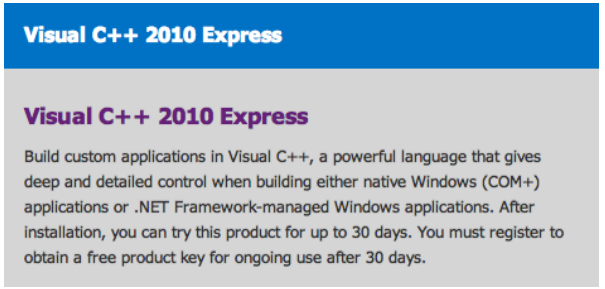
- ...is portable.
  - As one of the most frequently used languages in the world and as an open language, C++ has a wide range of compilers that run on many different platforms that support it. Code that exclusively uses C++'s standard library will run on many platforms with few to no changes.
- ...is upwards compatible with C
  - C++, being a language that directly builds off C, is compatible with almost all C code. C++ can use C libraries with few to no modifications of the libraries' code.
- ...has incredible library support.
  - A search for "library" on the popular project-management website SourceForge will yield over 3000 results for C++ libraries.

Programming environment

- Tools
  - Editor
  - Compiler: translate the source code to machine code to be executed
  - Interpreter: reads a little source code, translates it to machine code, and executes it, than reads a little more, etc.
  - Debugger: helps you step through code, shows you variables and flow of execution
- Interfaces / components / libraries
  - Linker: connects code from libraries with your code to make one executable
- Integrated Development Environments

IDEs for C++

- Windows:
  - Microsoft Visual C++
  - http://www.visualstudio.com/



  - http://www.mingw.org/
  - HOWTO: http://www.mingw.org/wiki/HOWTO_Install_the_MinGW_GCC_Compiler_Suite

IDEs for C++

- Linux
  - Eclipse CDT
  - http://www.eclipse.org/cdt/

http://en.wikipedia.org/wiki/Comparison_of_integrated_development_environments#C.2FC.2B.2B

Hello World!

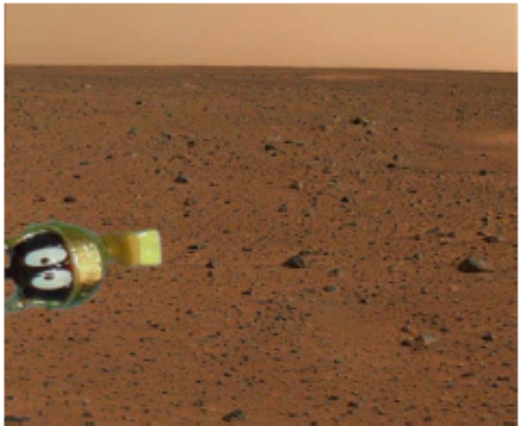
```cpp
// my first program in C++
#include <iostream>

int main()
{
  std::cout << "Hello World!";
}
```

```
Hello World!
```

Task 1: Prepare your programming environment and run Hello World!

Why programming?

- Our civilization runs on software
  - Most engineering activities involve software
- Note: most programs do not run on things that look like a PC
  - a screen, a keyboard, a box under the table
- Aircrafts, ships, communication, phones, energy…
  - there's a lot more to computing than games, word processing, browsing, and spreadsheets!



Mars rover autonomous driving system (incl. scene analysis and route planning)
See http://www.stroustrup.com/applications.html

A first program – just the guts…

```
// ...

int main()              // main() is where a C++ program starts
{
    cout << "Hello, world!\n";      // output the 13 characters  Hello, world!
                                    // followed by a new line
    return 0;           // return a value indicating success
}

// quotes delimit a string literal
// NOTE: "smart" quotes " " will cause compiler problems.
//    so make sure your quotes are of the style  " "
// \n is a notation for a new line
```

A first program – complete

```
// a first program:

#include "std_lib_facilities_3.h"      // get the library facilities needed for now

int main()                    // main() is where a C++ program starts
{
    cout << "Hello, world!\n";  // output the 13 characters  Hello, world!
                               // followed by a new line
    return 0;                 // return a value indicating success
}

    // note the semicolons; they terminate statements
    // braces { ... } group statements into a block
    // main( ) is a function that takes no arguments ( )
    //      and returns an int (integer value) to indicate success or failure
```

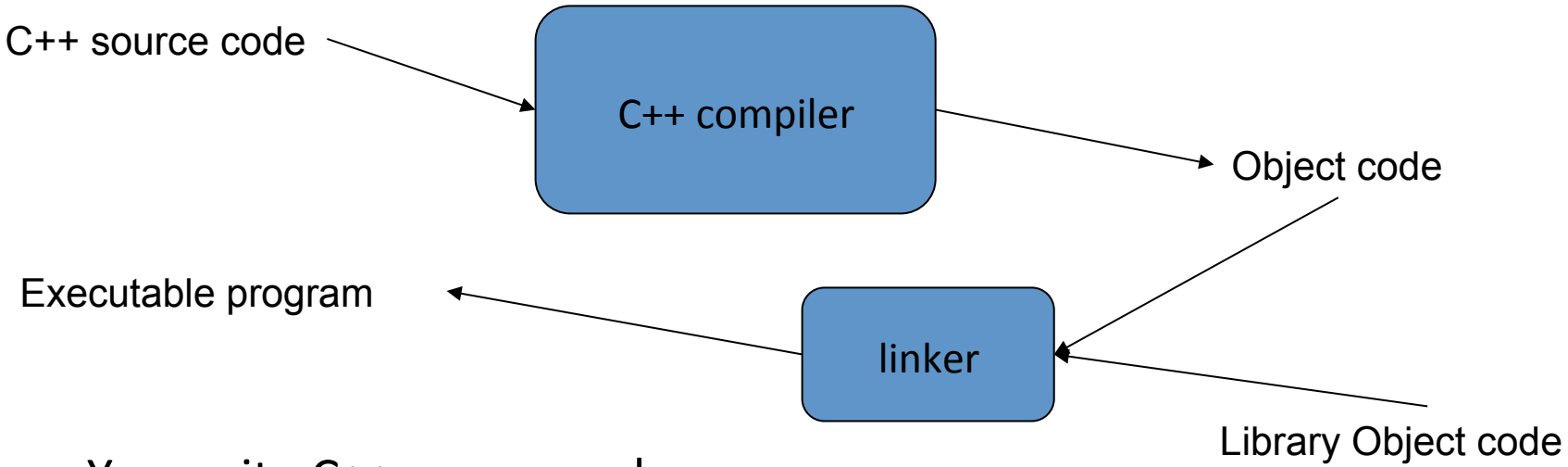
A second program

```
// modified for Windows console mode:

#include "std_lib_facilities_3.h"   // get the facilities for this course

int main()                  // main() is where a C++ program starts
{
    cout << "Hello, world!\n";      // output the 13 characters  Hello, world!
                                    // followed by a new line
    keep_window_open();       // wait for a keystroke
    return 0;                 // return a value indicating success
}

// without keep_window_open() the output window will be closed immediately
// before you have a chance to read the output (on Visual C++ 2003)
```

Hello world

- Only cout << "Hello, world!\n" directly does anything
- That's normal
  - Most of our code, and most of the systems we use simply exist to make some other code elegant and/or efficient
  - "real world" non-software analogies abound
- "Boiler plate," that is, notation, libraries, and other support is what makes our code simple, comprehensible, trustworthy, and efficient.
  - Would you rather write 1,000,000 lines of machine code?
- This implies that we should not just "get things done"; we should take great care that things are done elegantly, correctly, and in ways that ease the creation of more/other software: style matters!

Compilation and linking



- You write C++ source code
- The compiler translates what you wrote into object (or machine) code
- The linker links your code to system code needed to execute
  - E.g. input/output libraries, operating system code, and windowing code
- The result is an executable program
  - E.g. a .exe file on windows or an a.out file on Unix

So what is programming?

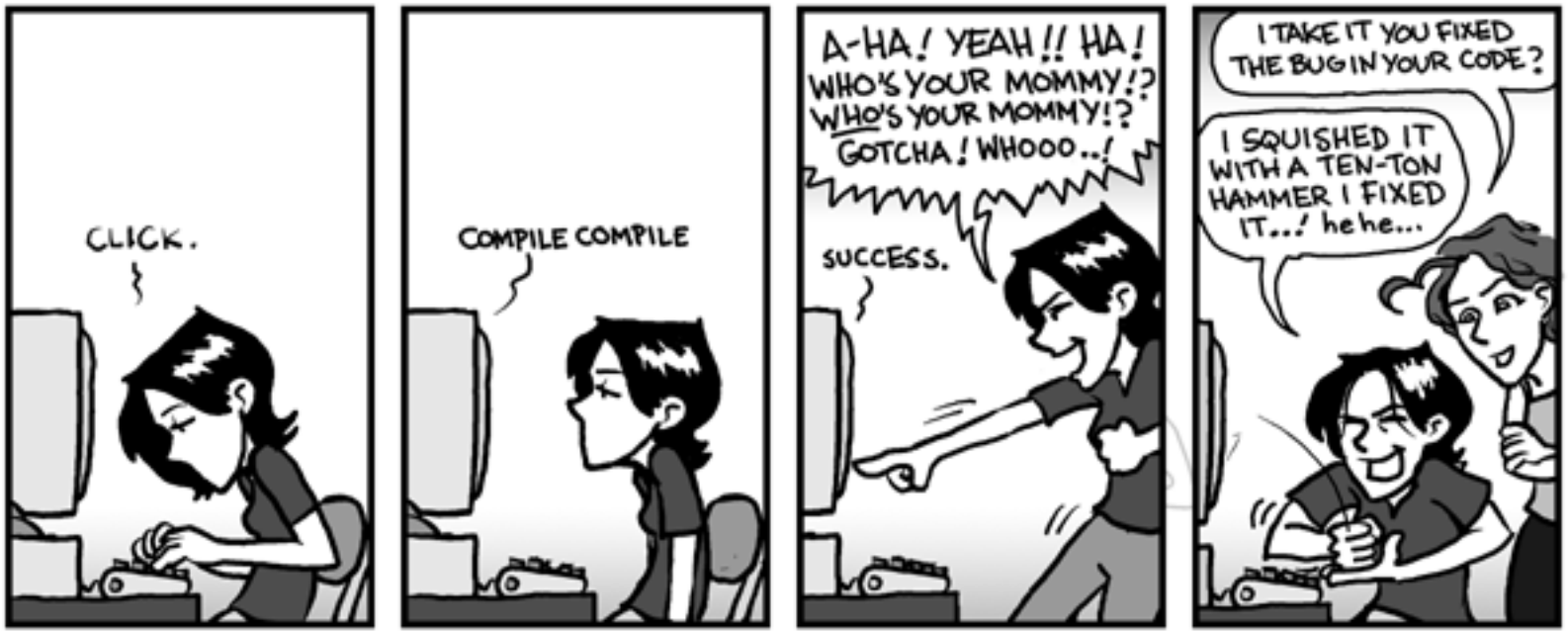
- Conventional definitions
  - Telling a very fast moron exactly what to do
  - A plan for solving a problem on a computer
  - Specifying the order of a program execution
    - But modern programs often involve millions of lines of code
    - And manipulation of data is central
- Definition from another domain (academia)
  - A … program is an organized and directed accumulation of resources to accomplish specific … objectives …
- Used here:
  - Specifying the structure and behavior of a program, and testing that the program performs its task correctly and with acceptable performance
- Software == one or more programs

Programming

- Programming is fundamentally simple
  - Just state what the machine is to do
- So why is programming hard?
  - We want "the machine" to do complex things
    - And computers are nitpicking, unforgiving, dumb beasts
  - The world is more complex than we'd like to believe
    - So we don't always know the implications of what we want
  - "Programming is understanding"
    - When you can program a task, you understand it
    - When you program, you spend significant time trying to understand the task you want to automate
  - Programming is part practical, part theory
    - If you are just practical, you produce non-scalable unmaintainable hacks
    - If you are just theoretical, you produce toys

Let's do it. It is fun.

- Task 1: find yourself an editor, compiler and linker (IDE), your tools
- "Hello world" is a very important program. Its purpose is to help you get used to your tools. Type in the program carefully
  - After you get it to work make a few mistakes to see how the tools respond; for example
    - Forget the header
    - Forget to terminate the string
    - Misspell return (e.g. retrun)
    - Forget a semicolon
    - Forget { or }
    - …



http://pu.inf.uni-tuebingen.de/users/klaeren/epigrams.html