Practical Programming
in Python

Inspired by 'Practical Programming' by Paul Gries, Jennifer Campbell, Jason Montojo

Lecture 1
What is Programming?

Why computers & programming?, What is programming?, Who are the programmers?

Kurt Rinnert, Kate Shaw

Physics Without Frontiers

The Abdus Salam International Centre for Theoretical Physics

Abstract

"Specialization is for insects."
– Robert A. Heinlein

We will discuss why programming is useful, what programming is and what it is *not*.
There will also be some remarks on the people who do the programming – soon you'll be one of them.

Overview

- Computers & programs (software) are everywhere
- Programming enables you to "teach" a computer (even an old one!) new tricks
- Programming makes computers versatile
- Programming does *not* require you to "think" like a computer
- Programmers are just people writing programs

A lot of this will seem obvious. Explicitly stating the obvious is a programmer's virtue.

Computers & Software are Everywhere

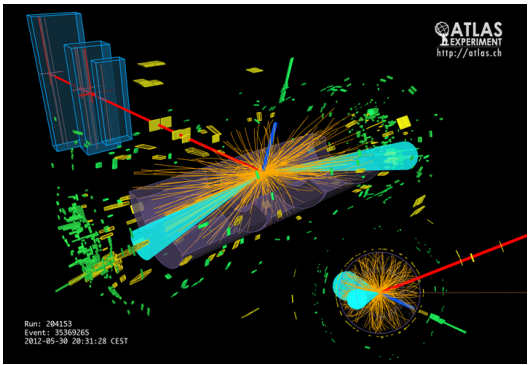
- Everyone would call these “watches”
- Inside, they *are* computers
- These two very likely share a lot of software
- Yet, there also must be differences in their programming

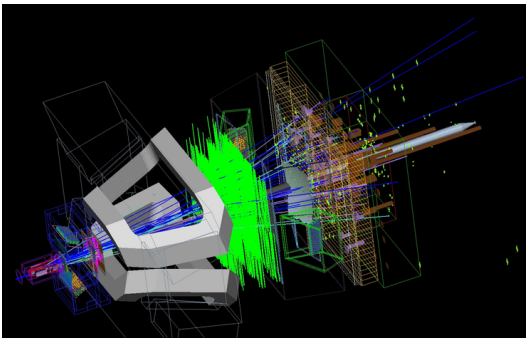
Computers don’t necessarily look like computers anymore.

And We Mean *Everywhere*

No space exploration without computers & software.

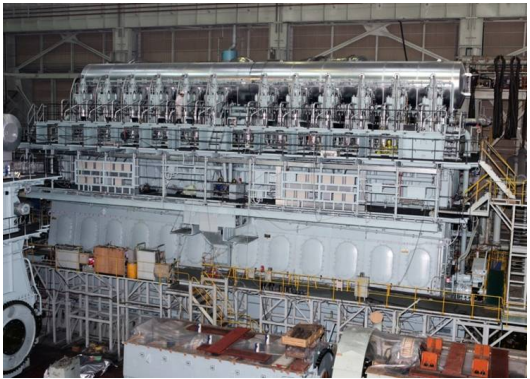
Huge Data Sets – High Energy Physics (HEP)





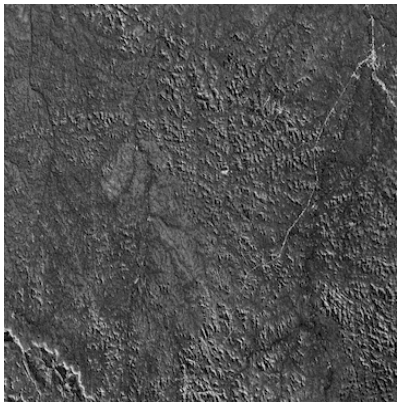
Acquiring, storing & analysing large volumes of data requires computers & programming.

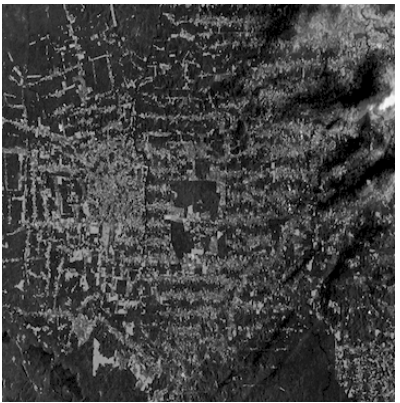
Design, Control & Monitoring



Modern engineering is unthinkable without computers & programming.

Quantifying Things





These are pictures of rain forest coverage. They clearly differ – but by how much *exactly*?

What is Programming?

- Define the problem precisely (this can't be overstated)
- Systematically analyse the problem, breaking it down into manageable parts
- Devise a solution
- Make sure the parts work (try to break them)
- Put it together
- Test your solution to the best of your abilities (try to break it)
- In case of failures (oh, there *will* be failures) repeat any of the above
- Decide when things are "good enough" pragmatically (listen to your clients)

Above all, programming is disciplined problem solving.

What Programming is *Not*

- Using a computer
- Writing code (although you'll have to do that)
- Changing examples and hoping for the best
- Knowing all details of a particular programming language
- Strictly following a paradigm
- Doing a perfect job (this is impossible)
- "Thinking" like a computer
- A way to become a lonely nerd

A lot can be done with pen a paper. Computers make many more things possible.

Programming Computers vs. Everyday Life

- Computers don't share human experiences
- This makes them really dumb in ways that come as a surprise to many people
- A programmer has to be aware of that
- How would you give a person directions from the dinner table to the bathroom?
  - The bathroom is upstairs, first door on the left
  - Oh, the light switch is on the right
- There a *many* assumptions making this work for your guest
- For instance, you can safely assume they won't try to walk over the table…

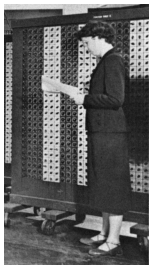
Programmers can't make assumptions that haven't been verified.

Who are the Programmers?

Kay McNulty

Fran Bilas

Ruth Teitelbaum



Betty Holberton

Jean Bartik

Marlyn Meltzer

The ENIAC programming team, the first regularly working programmers.

Who are the Programmers?

Ada Lovelace – the first programmer.

Grace Hopper – high level languages.

Margaret Hamilton – Apollo flight software.

Neil Armstrong would have had a tough time on the Moon without Margaret Hamilton.

Who are the Programmers?

- Not all programmers are computer scientists
- Not all computer scientists are programmers
- Some people don't like to be called programmers
- Programmers are simply people able to write programs
- As a programmer, you can
  - have fun
  - find new ways to reveal & quantify interesting things
  - save lives (eg. monitoring vaccination programs)
  - profoundly change your field of work
  - make your business more efficient
  - …

Richard Dawkins – biologist & self-professed programming addict (clean now, or so he claims).

Do not pay attention to stereotypes. Be proud of your skills.

Exercises Lecture 1