Welcome to MCS 260

MCS 260 Lecture 1
Introduction to Computer Science
Jan Verschelde, 11 January 2016

Introduction to Computer Science

Content of the Course

Text Books

Optional text books are

- J. Glenn Brookshear: *Computer Science. An Overview*. 10th edition, Addison Wesley, 2009 (or later edition).
- Bradley N. Miller and David L. Ranum: *Python Programming in Context*. Jones and Bartlett, 2009 (or later edition).

Two text books also good for the sequel to MCS 260:
MCS 275: Programming Tools and File Management.

Two other sequels to MCS 260 are

- MCS 320: Introduction to Symbolic Computation,
- MCS 360: Introduction to Data Structures.

Catalog Description for mcs 260

Computer literacy, number systems, concepts of operation systems, storage, files, databases, logic gates, circuits, networks, internet.

Introduction to programming in Python, variables, assignments, functions, objects.

Prerequisite(s): Credit or concurrent registration in MATH 180.

Computer Science. An Overview
J. Glenn Brookshear

- offers a solid introduction to Computer Science
  - hardware: how a computer works
  - software: how to use a computer
- computer literacy
  - understand basic terminology
  - overview of a discipline
- coverage:
  - explicitly several essential materials
  - use implicitly as a general reference

Python Programming in Context

Bradley N. Miller and David L. Ranum

- concise treatment of a broad range of topics
- we will cover
  1. in first 9 lectures (till 1st midterm):
     data structures, control constructs, functions

     $\rightarrow$ programming in the small (chapters 1 to 5)
  2. lectures after the midterms (lectures 12 to 21):
     modules, manipulating files, classes and objects

     $\rightarrow$ programming in the large (chapters 10, 11, 12)
  3. video game development (chapter 13)

Introduction to Computer Science

Purpose of the Course

Expectations are twofold:

- Understand Basics of Computer Science
- Implement Algorithms in Python

Goals: computer literacy and introduction to programming.

Computer Science is an active discipline → learn by doing.

Therefore:

1. emphasis on four computer projects,
2. active participation to the lab sessions.

You must adhere to the Acceptable Use Policy of computer labs!
Observe `accc.uic.edu/policy/acceptable-use-policy`,
for instance: no eating and no drinking in the computer labs.

the software we will use

We use free and open source sofware:

- Python version 3.5 (`www.python.org`):
  - We do not use any version of Python less than 3.
  - The Integrated Development Environment *IDLE* may need a separate installation.

Optionally, there is Sage (`www.sagemath.org`)
and the Sage Cell server at `https://sagecell.sagemath.org`.

The Sage cell server allows you to execute Python code
from within a web browser.

Introduction to Computer Science

What is Computer Science?

Chapter 0 of Computer Science. An Overview

from the Association of Computing Machinery (ACM):

> *“systematic study of those algorithms which describe and transform information: the underpinning theory, analysis, planning, efficiency, realization and application”*

ingredients in definition point at three different subfields:

1. theory and analysis: mathematics
2. planning, efficiency, and realization: engineering
3. applications: business

The concept of an algorithm

example: compute gcd(a, b), for $a, b \in \mathbb{N}$, $a \geq b > 0$.
Euclid's algorithm for the greatest common divisor

```
let $r$ be the remainder of $a/b$;
if $r = 0$
 then $b$ is the gcd;
 else the gcd is gcd($b, r$);
end if.
```

Like a recipe, the main qualities of an algorithm are clarity, correctness, and efficiency.

An algorithm is an *ordered* set of *unambiguous*, *executable* steps that define a *terminating* process.

Pseudocode: writing an algorithm in our own words

What is pseudocode?

- definition of an algorithm in a natural language,
- intended for human reading, but sufficiently detailed for coding.

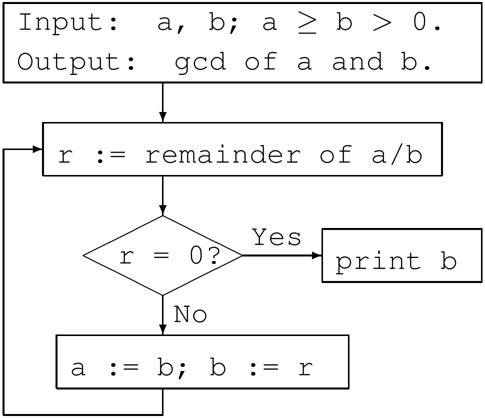
How to write good pseudocode?

- Start with an input/output specification
  input: what is given, output: what is computed.
- Name that variables that will be manipulated.
- Define the sequence of instructions.
- Some instructions are subject to conditions:

  ```
  if some condition holds
   then do this,
   else do that.
  ```
- Some instructions must repeated, for example:
  - while some condition holds do this,
  - repeat this until some condition is satisfied.

Flowchart

pictures of algorithms



Languages for Programming Algorithms

a quick overview

FORTRAN	(formula translator) is listed as one of the top ten algorithms of the past century.
COBOL	is widely used for business applications.
Algol 60	led to Pascal and Ada, rigorously designed.
C, C++, and Java	are perhaps the most successful ones.
Lisp	is for functional programming.
Prolog	is based on mathematical logic.
Python	will be our first programming language.

Ada Lovelace

Augusta Ada King, Countess of Lovelace (1815-1852)

- As mathematician she wrote a diagram to compute the Bernoulli numbers by the Analytical Engine of Charles Babbage.
- The language Ada is named in honor of her.
- Ada Lovelace Day, 2nd Tuesday of October.

  Celebrating the achievements of women in science, technology, engineering, and maths.

Portrait by Margaret Sarah Carpenter, downloaded from wikipedia.

Introduction to Computer Science

Introduction to Python

Why Python?
programmers prefer rapid application development tools

A Python session:

```
>>> def gcd(a,b):
...     r = a%b
...     if r == 0:
...         return b
...     else:
...         return gcd(b,r)
...
>>> gcd(12,8)
4
```

Overview of Python

history and features

Python is a free, open-source, general-purpose, interpreted, and scripting language for web applications.

History:	development started in late 1989 Python 2 is the legacy version. Python 3 is the present and future version.
Features:	easy, scalable, high level, object oriented, interpreted, extensible and flexible, rich core library, manages memory, support web applications, object distribution, databases, GUI programming, extendable and embeddable, exception handling, portable, freeware.

Visit www.python.org for documentation and resources.

Application Areas & Comparisons

Python is mainly used

- to create prototypes of an application,
- to glue together large software applications,
- to write CGI scripts for the internet.

Compared to other languages, Python

- is slower than compiled languages,
- is a good tool to test C and C++ programs,
- has nice syntax, nicer than other languages.

JPython is Python interpreter, written in Java.

Given Python code, Cython generates C code.

Introduction to Computer Science

The three big M's
commercial mathematical software

Commercial systems for mathematical and scientific computing are

Maple:	a computer algebra software system, computer algebra implements algorithms in symbolic and numerical computation.
Mathematica:	another major computer algebra system with a notebook interface and a logical programming language.
MATLAB:	a system for scientific computing, its main original strength is numerical linear algebra.

All these systems implement a wide variety of algorithms and provide support for high level data structures.
These systems allow to test new algorithms very quickly as their programming capabilities are easy to use.

Introduction to Computer Science

Open Source: Sage

Software for Algebra and Geometry Experimentation

Sage is a free open source mathematics software system with a Python-based language available at www.sagemath.org.

The mission of Sage is to create a viable free open source alternative to Magma, Maple, Mathematica, and MATLAB.

- Sage can be used to study a huge range of mathematics.
- Sage includes interfaces to Magma, Maple, Mathematica, MATLAB, and MuPAD, and the free programs Axiom, GAP, GP/PARI, Macaulay2, Maxima, Octave, and Singular.
- Sage can be used from your web browser either connecting to programs running on your computer or running somewhere else.

Sage is free software distributed under the GNU GPL license.

Summary

In this lecture we covered

- *Computer Science*: chapter 0 and
  section 5.1: the concept of an algorithm;
  We define algorithms with words by pseudocode.
  We define algorithms with pictures by flowcharts.
- the introduction of *Python Programming*.
  We use Python 3.5, which is *free*, visit www.python.org.

On Tuesday or Thursday, go to lab SEL 2263

- make sure your netid is working,
- there will be a quiz at the end.

Assignments

In the lab sessions of this week, consider the following problems:

1. Explain the differences between formulas and algorithms. What are the similarities?
2. What is the difference between a heuristic and an algorithm? Illustrate the difference with an example.
3. Write down the instructions to walk from BSB 389 to SEL 2263.
4. Describe (*give pseudocode for*) the algorithm to compute $a + b$, with a and b given as sequences of digits.
5. Do the previous exercise for the multiplication $a \star b$.
6. Draw (*give a flowchart for*) the algorithm to compute $a + b$, with a and b given as sequences of digits.
7. Do the previous exercise for the multiplication $a \star b$.

A selection of these homework problems will be collected and graded.