CS17700

Hello world

Instructors

- Voicu Popescu, instructor
  - popescu@purdue.edu
  - http://www.cs.purdue.edu/homes/popescu/
- Lorenzo Martino, instructional coordinator
  - lmartino@purdue.edu
- Teaching assistants

CS17700

- Two major goals
  - An introduction to Computer Science principles
  - An introduction to Computer Science practice using Python
- Targeted audience characteristics and needs
  - No Computer Science background
  - Collaboration with computer scientists
  - Use of complex computer science tools (i.e. software)
  - Development of custom computer science tools

Course organization

- Webpage
  - http://wiki.cs.purdue.edu/177
- Lecture
  - New concepts are introduced
- Recitation
  - Concepts are explained in more detail, reinforced
- Lab
  - Concepts are practiced

Communication

- Piazza ("marketplace" in Italian)
  - Online forum where students post questions and students and instructors post answers
  - Better scalability then direct, one to one email
  - Instructions posted on class webpage
  - Policies
    - Do not post lab or project solutions, partial solutions, incorrect solutions (cheating)
    - Use #private tag if not sure
    - Make questions general, clear, and concise

Communication

- Piazza
- Office hours
  - See webpage for details
  - Not a substitute for recitation or labs
- Instructor available after class for questions
  - I'll stay as long as needed (hallway if need be)
- In class via iClicker

Syllabus overview

- Computer Science Principles: ~6 weeks
  - Data, data structures, introduction to algorithms, basic algorithms, recursion
- Computer Science Practice: ~6 weeks
  - Programming in Python (data structure implementation, control flow, functions, debugging, recursion, advanced data processing)
- Computing and society: ~3 weeks
  - Internet, cyber security, and societal impact of computing

Resources

- Slides
- Text book
  - Python Programming: An Introduction to Computer Science. John Zelle. Second Edition. Franklin, Beedle & Associates Inc.
- Wiki Book (online book)
  - http://en.wikibooks.org/wiki/Python_Programming

Grading

- Attendance of lectures and recitations
  - 5% of course credit
  - No attendance taken the first week
  - After 4 lectures missed, 1% off for every additional lecture absence
  - After 2 recitations missed, 1% off for every additional recitation absence
  - This includes all absences (e.g. interviews, conferences, short term health issues, work, etc.)

Grading

- Attendance of lectures and recitations
  - 5% of course credit
- Weekly lab
  - 25% of course credit
- Projects
  - 5 x 5% = 25% of course credit
  - Late policy
    - <24h -20% of project credit
    - >24h & <48h -50% of project credit
    - >48h -100% (no credit)
- Midterm examinations
  - 2 x 12.5% = 25% of course credit
- Final examination
  - 20% of course credit

Policies

- All CS 17700 students have to
  - Familiarize themselves with CS policies
    - http://spaf.cerias.purdue.edu/cpolicy.html
  - Confirm knowledge of and adherence to CS policies
    - http://www.cs.purdue.edu/
    - Log into CS Portal using Purdue Career credentials
    - Click on “Academic Integrity Policy” on the left tab
    - Read policies carefully
    - Logging in is equivalent to e-signature

CoS Teaming Requirement

- SCI 210
  - Principles of working in teams
  - Blackboard module, first 6 weeks of the semester
- Two or three CS 17700 team projects
  - Practice of working in teams
  - Project questions will evaluate understanding of teaming

Computer Science

A 35,000 feet flyover

Computers

- Malleable tools for processing data

Data

- “Factual information used as a basis for reasoning, discussion, or calculation” M. Webster
- Can be stored, transformed, and transmitted
- Examples
  - Names of people in this class
  - A self-portrait by Van Gogh
  - Results of a molecular dynamics simulation

Why process data?

- To derive insight and knowledge
- For entertainment
- Examples
  - Searching for evidence of extraterrestrial life in radio signals coming from space
  - Playing Wii Tennis

Computers process data *fast*

- High clock frequency
  - 1GHz CPU clock means that one add takes 1 billionth of a second
  - Moore's Law
    - Transistor density doubles approximately every 2 years
    - Affects speed (denser means shorter distances thus faster)
    - Technological barriers will increase doubling period to 3 years at the end of 2013
- Parallel processing
  - Multiple processors, each with multiple cores
  - Parallel programming is a fundamental problem in CS

Computers process data *accurately*

- Computer HW is accurate
  - No arithmetic errors
    - Almost none (Pentium FDIV bug caused division errors)
  - No memory or disk reading errors
    - Unless hardware failure

66MHz Intel Pentium with the FDIV bug

Computers process data *accurately*

- Computer HW is accurate
- Not to be confused with SW accuracy
  - SW can be wrong due to incorrect programming, incorrect input, malicious attacks, etc.
  - Very difficult to prove SW correctness
    - Can be done for small programs
    - Would preclude most important and fun applications
  - SW licenses defer liability
    - Unlike engineering products (e.g. cars, bridges)
    - Like medical services (e.g. “infection can occur”)

Computers process data *accurately*

- Computer HW is accurate
- Not to be confused with SW accuracy
- However, we should
  - Follow good practices when writing programs
  - Test programs
  - Specify how programs are to be used
  - Address problems when reported by our users

Computers process data *accurately*

- Computer HW is accurate
- Not to be confused with SW accuracy
- However, we should
  - Follow good practices when writing programs
  - Test programs
  - Specify how programs are to be used
  - Address problems when reported by our users
  - "Program correctness is not possible nor required, and Microsoft, Adobe, and Apple can't do it either" defense will not fly in CS 17700

Computers excel at *low-level* data processing

- Computers can easily
  - Search through billions of words to find a given word
  - Increase the brightness in billions of images
  - Sort billions of health records alphabetically
- Computers have a harder time
  - Understanding natural language (e.g. humor, irony, sarcasm)
  - Deciding which of two paintings is better
  - Reconstructing the 3-D geometry of a real world scene from photographs
  - Not impossible, subject of ongoing research

How do computers process data?

- Data processing is described in algorithms
- Algorithm
  - A set of step-by-step instructions
  - Takes input data and produces output data in a finite amount of time
- Algorithms are encoded into programs to be understood and executed by computers
- Programs are written in programming languages

Programming languages

- At first, they were low level: machine code
  - “000000 00001 00010 00110 00000 100000” stands for add registers 1 and 2 and place the result in register 6
- Then higher level: assembly language
  - Introduction of mnemonics, or letter groups suggesting instruction name

Motorola MC6800 Assembly Language

```
                    ****************************************
                    * FUNCTION: INCH - Input character
                    * INPUT: none
                    * OUTPUT: char in acc A
                    * DESTROYS: acc A
                    * CALLS: none
                    * DESCRIPTION: Gets 1 character from terminal

C010 B6 80 04   INCH    LDA A  ACIA      GET STATUS
C013 47                 ASR A            SHIFT RDRF FLAG INTO CARRY
C014 24 FA              BCC    INCH      RECIEVE NOT READY
C016 B6 80 05           LDA A  ACIA+1    GET CHAR
C019 84 7F              AND A  #$7F      MASK PARITY
C01B 7E C0 79           JMP    OUTCH     ECHO & RTS
```

Programming languages

- At first, they were low level: machine code
- Then higher level: assembly language
- Now: high-level programming languages
  - English like instructions
  - Easier to program, to debug, to extend
  - Hardware (CPU) still executes machine code, thus need for compiler
    - Compiler translates program written in high-level language to machine code

High-level programming language example

```
sum = 0;
for(i = 0; i < 10; i++)
        if (a[i] > 0)
                sum = sum + a[i];
```

High-level programming language example

```
// this program computes the sum of the
//       positive numbers in an array of 10 numbers
sum = 0; // initialize the sum to 0
for(i = 0; i < 10; i++) // traverse the array, starting from
                        // first number, until the last, one at
                        // the time
        if (a[i] > 0) // if the current number is positive
                sum = sum + a[i]; // add it to the sum
```

Programming languages

- We will be using Python
  - High level
  - Lowest learning curve
  - It's a great time to start out in CS
    - No machine code or assembly

How do computers process data?

- Data processing is described in algorithms
- Algorithms are encoded into programs to be understood and executed by computers
- Programs are written in programming languages
- Programs are run on computers with the help of operating systems
  - Software that helps manage computer resources (memory, drives, mouse, display)
  - MacOS, Unix, Linux, Windows 7, Android, etc.

Remember first slide?

- Computers: malleable tools for processing data
  - We talked about data
  - About how computer process data
  - Malleable?
- Computer functionality is virtually infinite
  - New programs extend functionality
  - So far you have been using programs written by others
  - This course will teach you how to write you own programs