CSCI 2320
Principles of Programming Languages

Introduction

Mohammad T. Irfan
mirfan@bowdoin.edu
www.bowdoin.edu/~mirfan
Office Hours: MW 3–5 [till 4/22]
Office Room: VAC 309
Course Information

► Syllabus
► Grading, etc.

http://www.bowdoin.edu/~mirfan/CSCI-2320.html

Taking class notes

When the best way to take notes is by hand

Most students take a laptop with them to lectures. But are there times when they might be better off taking a pad and pen?
Handwritten class notes

- Deeper cognitive processing compared to typing
- Students can explain better
- It's hard to avoid typing verbatim even if you are asked not to

How to ace this course?

- Read the book. Lectures not enough.
- Stay organized
- Conceptualize theory with examples
- Start early with the assignments
  - Set aside a lot of time for debugging
- Ask lots of questions!
Difficulty level

- Understanding theoretical concepts
- Mastering recursion
- Debugging codes efficiently
- Adapting fast to code in new languages

What is a programming language?

- Beginning of computer programs
  https://www.youtube.com/watch?v=uBbVbqRvqTM
- Natural language vs. programming language
  - 2 key differences
Why this course?

- Understand how prog. languages work
- Express ideas better
- Learn new language fast
  - Tiobe index
- Better use of known languages
- Choose an “appropriate” language
  - Emergence of data science
- Better understanding of implementation

What will be covered?

- Two parts
  - Design principles
  - Paradigms
Design Principles (~8 weeks)

- Syntax
- Semantics
- Names and types
- Functions
- Memory management

Assignments 1, 2, 3: Implementation of some of these design principles

Compiler vs. interpreter

Paradigms (~6 weeks)

- Imperative
- Object-oriented programming (OOP)
- Web
- Functional

Bjarne Stroustrup:
https://www.youtube.com/watch?v=NvWTnlQZj4

Assignments 4, 5: Practical applications of diverging paradigms
Art of language design (book)

- Architecture
- Technical setting
- Standards
- Legacy systems

Goals of language design (book)

- Simplicity and readability
- Clarity about binding
- Reliability
- Good support
- Good library base
- Orthogonality
- Efficient implementation