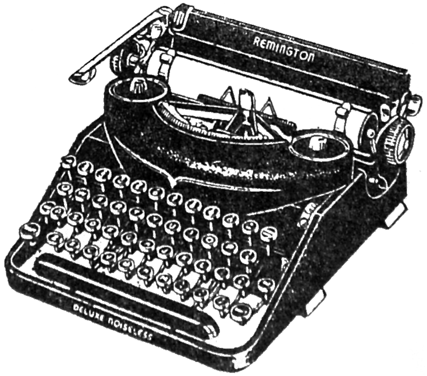
First steps in programming

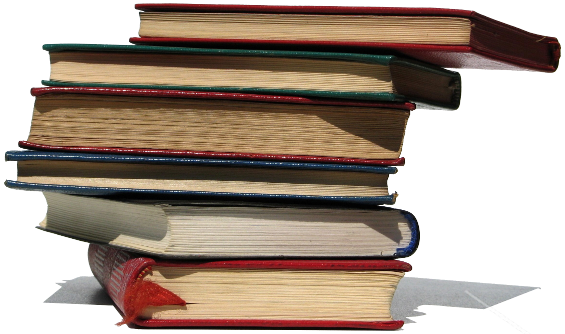
Jeffrey Wong

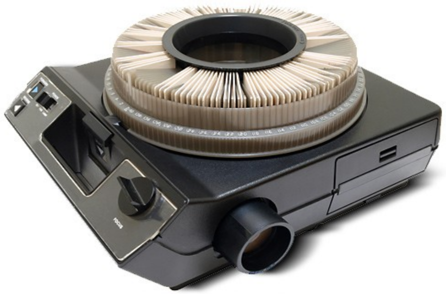

REMINGTON
DELUXE NOISELESS

RIKSTELEFON

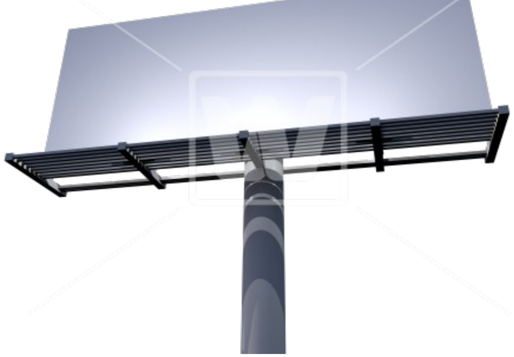
3M

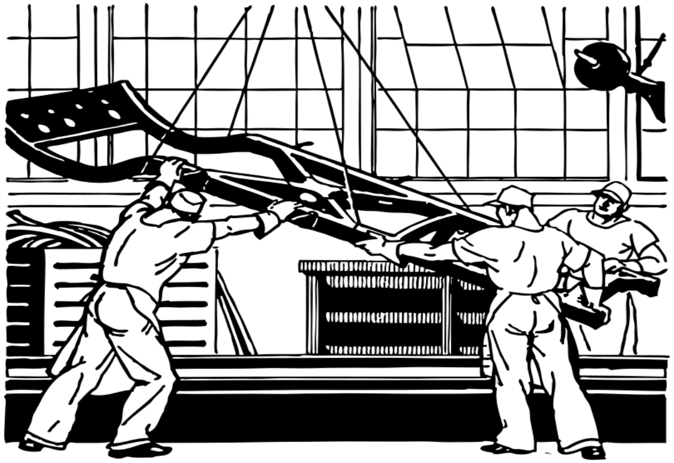
OneStep
POLAROID LAND CAMERA





SONY
AUTO REVERSE
20 PRESETS
AVLS
1
2
3
4
5
WALKMAN
WM-FX421
FM/AM





More than dollars and cents

Economic

- Competitiveness
- Widens sphere of interaction
- Widens entrepeneurial opportunities
- **You** are uniquely positioned in your field to sense
- Culture, limitations, opportunities

***Political**

- Awareness of how industry uses our information
- Awareness of how government uses information

Social

- Participate in our destiny versus having it “served” to us
- Awareness of bias and limitations of software/computers/machines

****Personal**

- Awareness of how humane and personal interaction are eroded

* “*Program or be programmed*” - David Rushkoff
**“*Who owns the future?*” - Jarome Lanier

Goal

- A gentle introduction to computer programming

Approach

Problem-solving

Mentored & Guided

Local and Interactive

What's out there?

Massive Online Course

- Coursera, Udacity, EdX, Treehouse

Classes

- HackerYou, pre-teen courses, Ladies learning code

Online

- Khan academy, instructional tutorial

Literature

- *Teach yourself, Dummy's Guide*

Issues/Opportunities

- **School**
- Most teachers are professional educators

- **Classes**
- Most geared toward web-development
- Those for children geared to gamification
- Over promising

- **MOOC**
- One-size fits all, impersonal
- Aimed at university-level
- Help/interaction *degrades* with increasing participation

- Literature
- Rarely didactic
- Typically geared toward software engineers...

What problems

Engineering/Robotics

- Manufacturing
- Medicine

Research/Prediction

- Weather
- Finance
- Disease

Arts

- Animation, design

News Media

- Data wrangling and visualization
- Digital research

Business

- Analytics
- Operations research/management

Advertising

- Search and search engine

Government

- Open data initiatives
- Forecasting
- Policy & planning

Education

- flipped-classroom

Future

- Data science
- Computational analytics

Criticisms

- A little knowledge is a dangerous thing
- Too much bad code already out there
- Not everyone wants or needs to be a software engineer
- There won't be enough jobs
- Coding will be an offshore activity

Wireframe

Time commitment

- 10 week course
- 2 meeting hours per week
- ~1 hour teaching
- ~1 hour questions and interaction

Learning objectives

- Emphasize comfort with programming tools
- Mathematical fundamentals
- Awareness of levels of programming abstraction
- Convey the power to manipulate data/information
-

Audience

- > 10 years? Basic mathematical literacy

Materials

- Computer

Python

- Elegant
- Abstract (but not too)
- Useful
- Basis for drilling down to C/C++ or bubbling up to Tornado, Javascript...

Scope

3 weeks

- What is a program
- Mathematical catch-up

3 weeks

- Programming basics

3 weeks

- Interweaved excercises
- Relevant and interesting
- Convey sence of accomplishment

Projects & Exercises

- Weekly challenge
- Process multiple csv documents
- Scrape a web page recursively

Tracking Progress

- Encourage documentation of accomplishments
- Blog as record of progress
- Blog as record of applied programming skills
- Track growth
- Encouragement

Detailed outline

Week 1

- Overview

Week 2

- Math & foundations

Week 3

- A computer script

Week 4

- Errors and debugging

Week 5

- Data types

Week 6

Control flow I

Week 7

Control flow II

Week 8

Functions

Week 9

Elementary web programming

Week 10

Final project

FINI

Research

Why is coding so hard to learn? (http://www.startuprob.com/learn-to-code/)

Lessons from students

- *Help sections aren't*
- not for beginners; one size fits all approach fails

- *Sudden leap in difficulty*
- critical threshold for progress; user-specific

- Inconsistent learning
- Use it or lose it

- Projects which aren't engaging
- Bored and unmotivated – what's in it for me?

- **Time spent by experts**
  - Fixing bugs
  - Adding features
  - Writing new software
  - Refactoring

- Overall tips
  - Commit
  - How to aid this process
  - Use a mentor
  - Could build up a network
  - Build your own projects
  - Review others' code

Research: Contacts

Resources: clients

- *City of Markham*
- *“Pre-teen” programs for Java, C++*
- *customerservice@markham.ca*

Resources for experts

- asd

Resources: Questionaire

- Demographics
- Age, sex, location
- Occupation or area of curiosity
- Motivation
- Why learn, why not, short/long goals
- Previous
- Approaches, materials, did you continue, why/obstacles
- Opinion
- Wish list – what to emphasize
- Unwish list – what to avoid

Business model

Fee

- $?

Typical profile

Self-motivated

- Open to self-directed learning
- NOT “social” activity

Curious

- Wishes to understand something for sake of understanding
- NOT always immediate, goal-oriented

Patient

- Find satisfaction in struggle
- NOT low threshold for frustration

Disciplined

- Work efficiently and regularly
- NOT easily distracted

Creative

- Ability to try their own idea
- NOT a copy paste approach