BBM 101 – Introduction to Programming I
Fall 2013, Lecture 1

Instructors: Aykut Erdem, Erkut Erdem, Fuat Akal
TAs: Ahmet Selman Bozkir, Gultekin Isik, Yasin Sahin, Pelin Aktas, Oguzhan Guclu
Today

■ Introduction
  ▪ About the class
  ▪ Organization of this course

■ What is computation?
  ▪ What is knowledge?
  ▪ What is a computer?
  ▪ What is a program?
  ▪ History of computing
Today

■ Introduction
  ▪ About the class
  ▪ Organization of this course

■ What is computation?
  ▪ What is knowledge?
  ▪ What is a computer?
  ▪ What is a program?
  ▪ History of computing
About the course

- Like most freshmen classes in Computer Science (CS) or Computer Engineering (CEng), this class will focus on solving problems and implementing their solutions as computer programs.
  - How to program

- However, you should know that CS and computer programming are not the same thing!

- So what is Computer Science?
What is Computer Science (CS)?

“Computer science is not really about computers – and it's not about computers in the same sense that physics is not really about particle accelerators, and biology is not about microscopes and Petri dishes ... and geometry isn't really about using surveying instruments.

Now the reason that we think computer science is about computers is pretty much the same reason that the Egyptians thought geometry was about surveying instruments: when some field is just getting started and you don't really understand it very well, it's very easy to confuse the essence of what you're doing with the tools that you use.”

Why study CS?

- **Its impact is broad and far-reaching.**
  - Internet. Web search, packet routing, distributed file sharing, ...
  - Biology. Human genome project, protein folding, ...
  - Computers. Circuit layout, file system, compilers, ...
  - Computer graphics. Movies, video games, virtual reality, ...
  - Security. Cell phones, e-commerce, voting machines, ...
  - Multimedia. MP3, JPG, DivX, HDTV, face recognition, ...
  - Social networks. Recommendations, news feeds, advertisements, ...
  - Physics. N-body simulation, particle collision simulation, ...

Adopted from: Sedgewick and Wayne
Why study CS?

- They may unlock the secrets of life and of the universe.
- Computational models are replacing mathematical models in scientific inquiry.

\[ E = mc^2 \]
\[ F = ma \]
\[ \left[ -\frac{\hbar^2}{2m} \nabla^2 + V(r) \right] \Psi(r) = E \Psi(r) \]

for (double t = 0.0; true; t = t + dt)
   for (int i = 0; i < N; i++)
      {  
         bodies[i].resetForce();
         for (int j = 0; j < N; j++)
            if (i != j)
               bodies[i].addForce(bodies[j]);  
      }

20\textsuperscript{th} century science
(formula based)

21\textsuperscript{st} century science
(algorithm based)

“Algorithms: a common language for nature, human, and computer.”
— Avi Wigderson

Adopted from: Sedgewick and Wayne
Why study CS?

- For fun and profit.

Adopted from: Sedgewick and Wayne
About this course

- This course serves as an introduction to the fundamentals of computer science and programming.
- The class will use the C programming language as a medium to provide a basic understanding of basic concepts in computer science.

Requirements

- Little or no programming experience!
- Solid high school algebra and a reasonable aptitude for mathematics

BBM 103 Introduction to Programming Practicum

- The students will gain hand-on experience via a set of programming assignments.
About this course (cont’d.)

■ Goals of the course:
  - What is computation?
  - The role of computation

■ Skills to develop:
  - Computational thinking
  - Understand codes
  - Understand abilities and limits
  - Maps problems into computation

■ What is computation?
  - What does it mean to think like a computer scientist?
  - Computational problem solving
BBM 101-103 Team

Instructors

Aykut ERDEM (Section 1)
Erkut ERDEM (Section 2)
Fuat AKAL (Section 3)

TAs

Ahmet Selman BOZKIR
Gultekin ISIK
Yasin SAHIN

Pelin AKTAS
Oguzhan GUCLUS

Office hours: to be determined
Reference Book

Communication

- The course webpage will be updated regularly throughout the semester with lecture notes, programming assignments and important deadlines.


BBM 101 - Introduction to Programming I, Fall 2013
Lectures: Wednesdays 13:00-15:45 @D1 (Sec. 1), @D2 (Sec. 2), @D4 (Sec. 3)
Practicum (BBM103): Thursdays 09:00-10:45 (Sec 01-03), 14:00-15:45 (Sec. 04-05)

“When human beings acquired language, we learned not just how to listen but how to speak. When we gained literacy, we learned not just how to read but how to write. And as we move into an increasingly digital reality, we must learn not just how to use programs but to make them.

In the emerging, highly programmed landscape ahead, you will either create the software or you will be the software. It’s really that simple: Program, or be programmed. Choose the former, and you gain access to the control panel of civilization. Choose the latter, and it could be the last real choice you get to make.”

~Douglas Rushkoff, in Program or Be Programmed: Ten Commands for a Digital Age (2010)

Instructors:
- Aykut Erdem (Sec. 1)
  - aykut-at-cs.hacettepe.edu.tr
  - Office: 111, Tel: 297 7500 / 146
  - TBA
- Erkut Erdem (Sec. 2)
  - erkut-at-cs.hacettepe.edu.tr
  - Office: 114, Tel: 297 7500 / 149
  - TBA
- Fuat Akal (Sec. 3)
  - akal-at-cs.hacettepe.edu.tr
  - Office: 201, Tel: 297 7500 / 129
  - TBA

Lectures:
- Wednesday, 13:00-15:45 @D1 (Sec. 1), @D2 (Sec. 2), @D4 (Sec. 3)

Practicum (BBM103):
- Thursdays 09:00-10:45 (Sec 01-03), 14:00-15:45 (Sec. 04-05)

TA:
- Ahmet Selman Bozkir
  - selman-at-cs.hacettepe.edu.tr
  - Office: 218, Tel: 297 7500 / 124
  - Office Hours: TBA

- Gultekin Isik
  - gultekin-at-cs.hacettepe.edu.tr
  - Office: 218, Tel: 297 7500 / 130
  - Office Hours: TBA

- Pelin Akgas
  - pelin-at-cs.hacettepe.edu.tr
  - Office: 218, Tel: 297 7500 / 130
  - Office Hours: TBA

- Oguzhan Gulsu
  - oguzhangan-at-cs.hacettepe.edu.tr
  - Office: 218, Tel: 297 7500 / 123
  - Office Hours: TBA

Course Description:
This course serves as an introduction to the fundamentals of computer science and programming. It aims to help students with little or no programming experience to gain necessary skills to work with abstract notions for solving computational problems. The course is structured around basic topics such as sequential, selective and repetitive structures, functions, recursion, arrays and pointers, structures and unions, basic input and output and strings. The class will use the C programming language as a medium to provide a basic understanding of basic concepts in computer science, and the students will gain hand-on experience via a set of programming assignments supplied in the complementary BBM 103 Introduction to Programming Practicum.

Reference books:
Getting Help

- **Office hours**
  - See webpage for the schedule

- **BBM 103 Introduction to Programming Practicum**
  - Course related recitations, practice with example codes, etc.

- **Communication**
  - Announcements and course related discussions through Piazza
    - BBM 103: [https://piazza.com/hacettepe.edu.tr/fall2013/bbm103](https://piazza.com/hacettepe.edu.tr/fall2013/bbm103)
Course work and grading

■ 3 midterm exams (10 + 30 + 15 = 55%)
  ▪ Closed book and notes
  ▪ In class on October 30th, November 20th, December 11th, respectively.

■ Final exam (40%)
  ▪ Closed book and notes
  ▪ To be scheduled by Registrar

■ Class participation (5%)
  ▪ Contribute to Piazza discussions
  ▪ Attend and participate in lectures
Course Overview

■ Part I
  ▪ Introduction *(0.5 week)*
  ▪ What is computation? *(0.5 week)*
  ▪ Binary representations *(0.5 week)*
  ▪ The Von Neumann architecture *(0.5 week)*

■ Part II
  ▪ Sequential structure *(1 week)*
  ▪ Selective structure *(1 week)*
  ▪ Repetitive structure *(1 week)*
  ▪ Functions *(1.5 weeks)*
  ▪ Pointers and Arrays *(2 weeks)*
  ▪ Structures *(1 week)*
  ▪ Input and Output *(0.5 week)*
  ▪ Strings *(0.5 week)*

Midterm exam #1
Midterm exam #2
Midterm exam #3
BBM 103 Introduction to Programming Practicum

- Programming assignments (PAs)
  - Five assignments throughout the semester.
  - Each assignment has a well-defined goal such as solving a specific problem.
  - You must work alone on all assignments stated unless otherwise.

- Important Dates (Tentative)
  - PA 1: October 9th
  - PA 2: November 6th
  - PA 3: November 20th
  - PA 4: December 4th
  - PA 5: December 18th
Policies

■ Work groups
  ▪ You must work alone on all assignments stated unless otherwise

■ Submission
  ▪ Assignments due at 23:59 on Wednesday evening
  ▪ Electronic submissions (no exceptions!)

■ Lateness penalties
  ▪ Get penalized 10% per day
  ▪ No late submission later than 3 days after due date
Cheating

■ What is cheating?
  - Sharing code: by copying, retyping, looking at, or supplying a file
  - Coaching: helping your friend to write a programming assignment, line by line
  - Copying code from previous course or from elsewhere on WWW

■ What is NOT cheating?
  - Explaining how to use systems or tools
  - Helping others with high-level design issues

■ Penalty for cheating:
  - Helping others with high-level design issues
  - Removal from course with failing grade

■ Detection of cheating:
  - We do check
  - Our tools for doing this are much better than most cheaters think!
Always keep in mind

- This is your **first** CS course.
- In order to get the most out of the course, **try to stay ahead**.
  - By the weekend, make sure you have reviewed the material covered in the lectures of the preceding week.
- If you have any questions or just want to chat about the course, do not hesitate to **go to office hours**.
- **Nothing is easy, but don’t give up!**
We aim high!

- Our goal is to make BBM 101 one of the **highly respected** and one of the **most enjoyable** courses in the department!

Photo courtesy of Flickr user richardlamprecht
Today

■ Introduction
  ▪ About the class
  ▪ Organization of this course

■ What is computation?
  ▪ What is knowledge?
  ▪ What is a computer?
  ▪ What is a program?
  ▪ History of computing
What is knowledge?

- **Declarative knowledge**
  - Axioms (definitions)
  - Statements of fact

  “y is the square root of x if and only if y*y = x” does not help to find the square root!
What is knowledge?

- **Declarative knowledge**
  - Axioms (definitions)
  - Statements of fact

  "y is the square root of x if and only if y*y = x"

  *does not help to find the square root!*

- **Imperative knowledge**
  - How to do something
  - A sequence of specific instructions (what computation is about)

**Babylonian method**

Get x as an input

1. Begin with an arbitrary positive number \( y_0 \) (an initial guess)
2. If \( y_n^2 \approx x \), stop
   - Else let \( y_{n+1} = (y_n + x/y_n)/2 \) (use the arithmetic mean to approximate the geometric mean).
3. Repeat step (2)
What is knowledge? (cont’d.)

**Another example** – Estimating greatest common divisor (gcd)

**Declarative definition**
“d is the gcd of a and b if and only if d is the largest possible integer satisfying a = d*x and b = d*y with x and y being two positive integers”

**Imperative definition: Euclid’s formula**
Get 2 positive integers a, b, a >= b as input
1. Divide a by b, call the remainder R
2. If R=0, stop
   Else let a = b and b = R
3. Repeat step ii

Use Euclid’s formula to compute gcd(48,18).
Then, what is a computer?

- A device that executes a sequence of computations and instructions.

- Modern computers are electronic and digital.

- Does pencil and paper count as a computer?
Programs

- These sequences of instructions and computations is called a program.

- We’ll be designing programs in this course.

- These programs will be based on algorithms.
When did the term ‘Computer’ originate?

- The definition from The Oxford Dictionary:
  
  "Computer (noun). A person who makes calculations, especially with a calculating machine."

![Image of a room filled with people working on machines.](image_url)

Courtesy of the Library of Congress
Fixed program computers

- Developed to solve a specific problem (set).

- Very old roots, old perspectives, ...
  - Abacus
  - Antikythera mechanism
  - Pascaline
  - Leibniz Wheel
  - Jacquard’s Loom
  - Babbage Difference Engine
  - The Hollerith Electric Tabulating System
  - Atanasoff-Berry Computer (ABC)
  - Turing bombe
  - etc.
Abacus (500 BC)

- First pocket calculator
- Still used by businessmen in Asia.
Antikythera mechanism (100 BC)

- First analog computer
- An ancient mechanical computer designed to calculate astronomical positions

© Rien van de Weygaert

© Antikythera Mechanism Research Project
Pascaline (1642)

- Blaise Pascal, 1642
- A mechanical calculator for performing two arithmetic operations: addition and subtraction

© Mark Richards
Leibniz Wheel (1694)

- Gottfried Wilhelm von Leibniz, 1694
- A mechanical calculator for performing all four arithmetic operations: addition, subtraction, multiplication and division

Courtesy of the Deutsches Museum, München
Jacquard’s Loom (1801)

- Developed in 1801 by Joseph-Marie Jacquard.
- The loom was controlled by a loop of punched cards.
- Holes in the punched cards determined how the knitting proceeded, yielding very complex weaves at a much faster rate.

A Jacquard Loom workshop - Germany, 1858.

Adopted from: T. Cortina
Babbage Difference Engine (1832)

■ Charles Babbage, 1832

■ A mechanical calculator designed to tabulate polynomial functions (can be used for solving polynomial equations, curve fitting, etc.)

■ A working difference engine was built in 1991 to celebrate the 200th anniversary of Babbage's birth (London Science Museum).

■ It could hold 8 numbers of 31 decimal digits each and could thus tabulate 7th degree polynomials to that precision.
The Hollerith Electric Tabulating System

- **1880 Census.** Took 1,500 people 7 years to manually process data.
- **Herman Hollerith.** Developed counting and sorting machine to automate.
  - Use punch cards to record data (e.g., gender, age).
  - Machine sorts one column at a time (into one of 12 bins).
  - Typical question: how many women of age 20 to 30?

- **1890 Census.** Finished months early and under budget!

Adopted from: Sedgewick and Wayne
Modern Punch Cards

- Punch cards. [1900s to 1950s]
  - Also useful for accounting, inventory, and business processes.
  - Primary medium for data entry, storage, and processing.

- Hollerith's company later merged with 3 others to form Computing Tabulating Recording Corporation (CTRC); the company was renamed in 1924.

Adopted from: Sedgewick and Wayne
Modern Punch Cards

- Punch cards. [1900s to 1950s]
  - Also useful for accounting, inventory, and business processes.
  - Primary medium for data entry, storage, and processing.

- Hollerith's company later merged with 3 others to form Computing Tabulating Recording Corporation (CTRC); the company was renamed in 1924.

IBM 80 Series Card Sorter, 1949
(650 cards per minute)

Adopted from: Sedgewick and Wayne
Atanasoff-Berry Computer (ABC) (1939)

- John Vincent Atanasoff and Clifford Berry, 1939-1942
- One of the first electronic digital computing devices
- Designed to solve a system of linear equations
Turing bombe (1941)

- Alan Turing, 1939
- Developed to crack German Enigma codes during WW II.
Alan Turing

- 1912-1954
- Considered the “father” of modern computer science.
- Presented formalisms for the notions of computation and computability in the 1930’s.
- Worked at Bletchley Park in Great Britain during WWII to develop Colossus to help break the German Enigma Code.
- Developed the notion in 1950 of a test for machine intelligence now called the Turing Test.

- The Turing Award, the highest award in computing, is named in honor of Alan Turing.
Stored Program Computers

- Problem solving

- What if input is a machine (description) itself?
- Universal Turing machines
  - An abstract general purpose computer
Universal Turing Machines

- **Tape.**
  - Stores input, output, and intermediate results.
  - One arbitrarily long strip, divided into cells.
  - Finite alphabet of symbols.

- **Tape head.**
  - Points to one cell of tape.
  - Reads a symbol from active cell.
  - Writes a symbol to active cell.
  - Moves one cell at a time.

Adopted from: Sedgewick and Wayne
Universal Turing Machines

- **Tape.**
  - Stores input, output, and intermediate results.
  - One arbitrarily long strip, divided into cells.
  - Finite alphabet of symbols.

- **Tape head.**
  - Points to one cell of tape.
  - Reads a symbol from active cell.
  - Writes a symbol to active cell.
  - Moves one cell at a time.

- Is there a more powerful model of computation? No!
  - Most important scientific result of 20th century?

Adopted from: Sedgewick and Wayne
Questions about computation

■ What is a general-purpose computer?
■ Are there limits on the power of digital computers?
■ Are there limits on the power of machines we can build?

David Hilbert  Kurt Gödel  Alan Turing  Alonzo Church  John von Neumann

Adopted from: Sedgewick and Wayne
Church-Turing thesis (1936)

Turing machines can compute any function that can be computed by a physically harnessable process of the natural world.

**Remark.** "Thesis" and not a mathematical theorem because it's a statement about the physical world and not subject to proof.

**Implications.**
- No need to seek more powerful machines or languages.
- Enables rigorous study of computation (in this universe).

**Bottom line.** Turing machine is a simple and universal model of computation.
Church-Turing thesis: evidence

- 8 decades without a counterexample.
- Many, many models of computation that turned out to be equivalent.

<table>
<thead>
<tr>
<th>model of computation</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enhanced Turing machines</td>
<td>multiple heads, multiple tapes, 2D tape, nondeterminism</td>
</tr>
<tr>
<td>untyped lambda calculus</td>
<td>method to define and manipulate functions</td>
</tr>
<tr>
<td>recursive functions</td>
<td>functions dealing with computation on integers</td>
</tr>
<tr>
<td>unrestricted grammars</td>
<td>iterative string replacement rules used by linguists</td>
</tr>
<tr>
<td>extended L-systems</td>
<td>parallel string replacement rules that model plant growth</td>
</tr>
<tr>
<td>programming languages</td>
<td>Java, C, C++, Perl, Python, PHP, Lisp, PostScript, Excel</td>
</tr>
<tr>
<td>random access machines</td>
<td>registers plus main memory, e.g., TOY, Pentium</td>
</tr>
<tr>
<td>cellular automata</td>
<td>cells which change state based on local interactions</td>
</tr>
<tr>
<td>quantum computer</td>
<td>compute using superposition of quantum states</td>
</tr>
<tr>
<td>DNA computer</td>
<td>compute using biological operations on DNA</td>
</tr>
</tbody>
</table>

Adopted from: Sedgewick and Wayne
Babbage’s Analytical Engine (1834, 1836)

- Designed around 1834 to 1836
  - was to be a universal machine capable of any mathematical computation
  - embodies many elements of today’s digital computer
  - a control unit with moveable sprockets on a cylinder that could be modified
  - separated the arithmetic operations (done by the mill) from the storage of numbers (kept in the store)
    - store had 1000 registers of 50 digits each
    - Babbage incorporated using punched cards for input
      - idea came from Jacquard loom
- Never built by Babbage due to lack of funds and his eventual death in 1871

Adopted from: T. Cortina
Ada Lovelace

- 1815-1852
- Daughter of poet Lord Byron

- Translated Menabrea’s Sketch of the Analytical Engine to English
  - Quadrupled its length by adding lengthy notes and detailed mathematical explanations

- Referred to as the world’s first programmer
  - Described how the machine might be configured (programmed) to solve a variety of problems.
The Zuse Z3 Computer (1941)

- Konrad Zuse, 1941
- The original Z3 was destroyed in a bombing raid of Berlin in 1943.
- Zuse later supervised a reconstruction of the Z3 in the 1960s (currently on display at the Deutsches Museum in Munich)
Colossus Mark 1 (UK, 1944)

- The world's first electronic digital computer with programmability.
ENIAC (Mauchly and Eckert, USA, 1946)

- The first large-scale general-purpose electronic computer without any mechanical parts.
- Designed to calculate artillery firing tables for the United States Army's Ballistic Research Laboratory.
EDVAC (von Neuman, USA, 1951)

- Unlike the ENIAC, it uses binary rather than decimal numbering system.
- Instructions were stored in memory sequentially with their data.
- Instructions were executed sequentially except where a conditional instruction would cause a jump to an instruction someplace other than the next instruction.

Adopted from G. Kesden
The Computer Tree


THE COMPUTER TREE

Adopted from: T. Cortina
Next week

- More on stored program computers
  - Von Neumann architecture
  - Components of a computer
- Binary representations
- Abstraction
- Programming paradigms
  - Programming languages
The Birth of the Computer

A TED talk given by George Dyson
http://www.ted.com/talks/george_dyson_at_the_birth_of_the_computer.html