Lecture 1:
Basic concepts for data structures
About the course

• This course will help students understand the **basic data structures** such as matrices, stacks, queues, linked lists, etc.

• **BBM 203 Programming Laboratory:** The students will gain hand-on experience via a set of programming assignments supplied as complementary.

• **Requirements:** You must know basic programming (i.e. BBM101).
References


• Data Structures Notes, Mustafa Ege.
The course web page will be updated regularly throughout the semester with lecture notes, programming assignments, announcements and important deadlines.

http://web.cs.hacettepe.edu.tr/~bbm201
Getting Help

• **Office hours**
  See the web page for details

• **BBM 203 Programming Laboratory**
  Course related recitations, practice with example codes, etc.

• **Communication**
  Announcements and course related discussions through
  BBM 201: [https://piazza.com/hacettepe.edu.tr/fall2020/bbm201](https://piazza.com/hacettepe.edu.tr/fall2020/bbm201)
  BBM 203: same as 201
Course Work and Grading

- 1 midterm exam (25%)
- Quizzes (25%) (5 out of 6)
- Final exam (50%)

The joy of learning
## Course Overview

### BBM201 Schedule (Tentative)

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Topic</th>
<th>Quiz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.10.2020</td>
<td>Orientation and Motivation</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>14.10.2020</td>
<td>Basic concepts for data structures, performance analysis, space and time complexity</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>21.10.2020</td>
<td>Representation of multidimensional arrays, matrix representation</td>
<td>Q1</td>
</tr>
<tr>
<td>4</td>
<td>28.10.2020</td>
<td>Record/struct, set</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4.11.2020</td>
<td>Linked list</td>
<td>Q2</td>
</tr>
<tr>
<td>6</td>
<td>11.11.2020</td>
<td>Stack, Queue</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>18.11.2020</td>
<td>Trees, Binary Trees, Binary Search Trees</td>
<td>Q3</td>
</tr>
<tr>
<td>8</td>
<td>25.11.2020</td>
<td>Midterm review, MIDTERM EXAM</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>2.12.2020</td>
<td>Balanced Trees</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9.12.2020</td>
<td>Hash Tables</td>
<td>Q4</td>
</tr>
<tr>
<td>11</td>
<td>16.12.2020</td>
<td>Graph representation</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>23.12.2020</td>
<td>String, Trie</td>
<td>Q5</td>
</tr>
<tr>
<td>13</td>
<td>30.12.2020</td>
<td>Heap, Priority Queue</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>6.1.2021</td>
<td>Final Exam Preparation</td>
<td>Q6</td>
</tr>
<tr>
<td>15-16</td>
<td></td>
<td>FINAL EXAM</td>
<td></td>
</tr>
</tbody>
</table>
• **Programming assignments (PAs)**
  • Four 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.
### BBM203 Schedule (Tentative)

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Lab</th>
<th>Assignment</th>
<th>Quiz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.10.2020</td>
<td>Tutorial: Java to C++ Transition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>15.10.2020</td>
<td>Tutorial: Java to C++ Transition</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>22.10.2020</td>
<td>Tutorial: Java to C++ Transition</td>
<td></td>
<td>Q1</td>
</tr>
<tr>
<td>4</td>
<td>29.10.2020</td>
<td>No Lab (Republic Day)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5.11.2020</td>
<td>Office hour</td>
<td>PA1: Array &amp; Matrices</td>
<td>Q2</td>
</tr>
<tr>
<td>6</td>
<td>12.11.2020</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>19.11.2020</td>
<td>Office hour, Recitation</td>
<td>PA2: Linked list</td>
<td>Q3</td>
</tr>
<tr>
<td>8</td>
<td>26.11.2020</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>3.12.2020</td>
<td>Office hour, Recitation</td>
<td>PA3: Stack &amp; Queue</td>
<td>Q4</td>
</tr>
<tr>
<td>10</td>
<td>10.12.2020</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>17.12.2020</td>
<td>Office hour, Recitation</td>
<td>PA4: Trees</td>
<td>Q5</td>
</tr>
<tr>
<td>12</td>
<td>24.12.2020</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>31.12.2020</td>
<td>Recitation</td>
<td></td>
<td>Q6</td>
</tr>
<tr>
<td>14</td>
<td>7.1.2021</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Policies

• **Work groups**
  • You must work alone on all assignments stated unless otherwise

• **Submission**
  • Assignments due at 23:59 (no extensions!)
  • Electronic submissions (no exceptions!)

• **Lateness penalties**
  • No late submission is accepted
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:
- Suspension from school for 6 months (minimum)

Detection of cheating:
- We do check: Our tools for doing this are much better than most cheaters think!
BASIC CONCEPTS 
FOR DATA STRUCTURES
Digital Data

Movies

Music

Photos

Protein Shapes

gatctttttta tttaaacagt ctctttatta gatcttttat taggatcatg atcctctgtg
gataagtgtat tatcatact gcagatcata taattaagga gatctggttg ttgtagtgga
cctggtcagc tattcgtatat aagcttggat ctaaatgcca ttgttatgcac agtcactcgg
cagaatcaag gttgttatgt ggatatctac tgggttttac cttgtttttaa gcataagttat
acacattcgt tcgogcgcct tttgacgtaa tttagatcata aaatccaat ctttgaccca

Maps

001010100101010100010010101000010010010100....
Data structures

- Primitive
  - Integer
  - Float
  - Characters
  - Pointers

- Non-primitive
  - Array
  - Lists
  - Files

- Non-linear
  - Tree
  - Graph

- Linear
  - Stack
  - Queue
Digital Data Must Be ...

- **Encoded** (e.g. 01001001 <-> 🎵)

- **Arranged**
  - Stored in an orderly way in memory / disk

- **Accessed**
  - Insert new data
  - Remove old data
  - Find data matching some condition

- **Processed**
  - Algorithms: shortest path, minimum cut, FFT, ...
How do we organize information so that we can find, update, add, and delete portions of it efficiently?
Data Structure Example Applications

• How does Google quickly find web pages that contain a search term?

• What’s the fastest way to broadcast a message to a network of computers?

• How can a subsequence of DNA be quickly found within the genome?

• How does your operating system track which memory (disk or RAM) is free?

• In the game Half-Life, how can the computer determine which parts of the scene are visible?
Suppose You’re Google Maps…

- You want to store data about cities (location, elevation, population)…

What kind of operations should your data structure(s) support?
Operations to support the following scenario…

Finding addresses on map?
- *Lookup city by name…*

Mobile user?
- *Find nearest point to me…*

Car GPS system?
- *Calculate shortest-path between cities…*
- *Show cities within a given window…*

Political revolution?
- *Insert, delete, rename cities*
How will you count user views on YouTube?

- Let's write a `userViewCount()` function

```c
int userViewCount (int current_count)
{
    int new_count;
    new_count = current_count + 1;
    return new_count;
}
```

Will this implementation work all the time?
How will you count user views on YouTube?

%99.9 times yes.
How will you count user views on YouTube?

YouTube's counter previously used a 32-bit integer

YouTube said the video - its most watched ever - has been viewed more than 2,147,483,647 times.

It has now changed the maximum view limit to 9,223,372,036,854,775,808, or more than nine quintillion.
How bad can it be?

- June 4, 1996
- Ariane 5 rocket launched by the European Space Agency
- After a decade of development costing $7 Billion (~42 Billion in Turkish Liras, just for comparison Istanbul’s third bridge cost estimates are 4.5 Billion TL)
- Exploded just 40 seconds after its lift-off
- The destroyed rocket and its cargo were valued at $500 million
- Reason?
How bad can it be?

- Reason?
- Inertial reference system error: specifically a 64 bit floating point number relating to the horizontal velocity of the rocket with respect to the platform was converted to a 16 bit signed integer.
- The number was larger than 32,767, the largest integer storable in a 16 bit signed integer, and thus the conversion failed.
- $500 Million rocket/cargo
- Time and effort
Floating Point Representation

Format of Floating points
IEEE754

64bit = double, double precision
1 11bit 52bit

32bit = float, single precision
1 8bit 23bit

16bit = half, half precision
1 5bit 10bit
# Floating Point Representation

## Nvidia Tesla Workstation GPU Performance Comparison

<table>
<thead>
<tr>
<th></th>
<th>P100</th>
<th>M40</th>
<th>K40</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Architecture</strong></td>
<td>Pascal</td>
<td>Maxwell</td>
<td>Kepler</td>
</tr>
<tr>
<td><strong>Double Precision (FP64)</strong></td>
<td>5.3 Tflop/s</td>
<td>0.2 Tflop/s</td>
<td>1.4 Tflop/s</td>
</tr>
<tr>
<td><strong>Single Precision (FP32)</strong></td>
<td>10.6 Tflop/s</td>
<td>7 Tflop/s</td>
<td>4.3 Tflop/s</td>
</tr>
<tr>
<td><strong>Half Precision (FP16)</strong></td>
<td>21.1 Tflop/s</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Memory Bandwidth</strong></td>
<td>720GB/s</td>
<td>288GB/s</td>
<td>288GB/s</td>
</tr>
<tr>
<td><strong>Memory Size</strong></td>
<td>16GB</td>
<td>12GB / 24GB</td>
<td>12GB</td>
</tr>
<tr>
<td><strong>Release Date</strong></td>
<td>2016</td>
<td>Nov-15</td>
<td>Nov-13</td>
</tr>
</tbody>
</table>
Goals

“I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships.”

Linus Torvalds, 2006
A data structure is a way to store and organize data in computer, so that it can be used *efficiently*.

Some of the more commonly used data structures include lists, *arrays*, *stacks*, *queues*, *heaps*, *trees*, and *graphs*.
What are data structures?

- Data structures are software artifacts that allow data to be stored, organized and accessed.

- Ultimately data structures have two core functions: put stuff in and take stuff out.
Why so many?

- Space efficiency
- Time efficiency:
  - Store
  - Search
  - Retrieve
  - Remove
  - Clone etc.
Choosing Data Structures

Queue vs Binary Tree

---Which one to use for what task?
Why So Many Data Structures?

- Ideal data structure:
  - “fast”, “elegant”, memory efficient

- Generates tensions:
  - time vs. space
  - performance vs. elegance
  - generality vs. simplicity
  - one operation’s performance vs. another’s

The study of data structures is the study of tradeoffs. That’s why we have so many of them!