Competitive Programming Lectures-3

Deniz Soylular / Enes Ak

Recap:

- Data Structures
    - Arrays
    - Sets
    - Stack and Queue
    - Hashmap

- Hash Function

KUding Contest!

- Will be held on **December 22-23-24**
- On **December 27** Award ceremony
- In total **17 problems,** 2 days 14 hours.

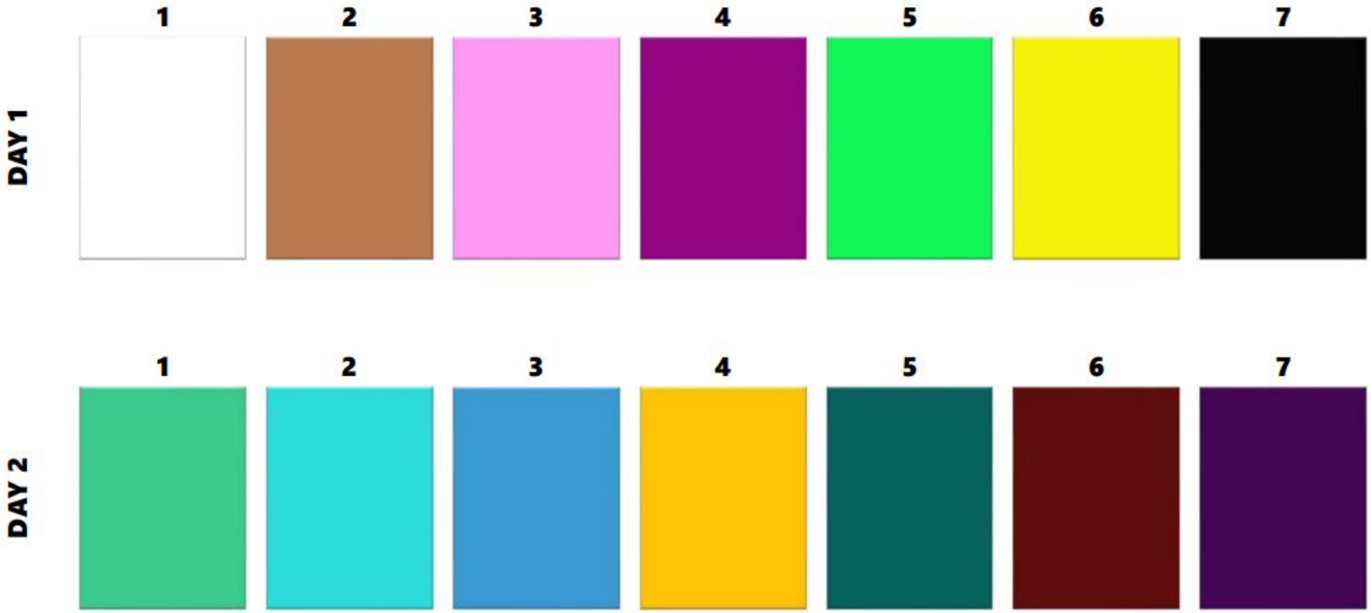
KU ACM

A Deeper Look Into the Format

- Only open to **Koç University students**
- The problems are **not intended to be overly challenging**
- Each problem set will be published **at 10 a.m.**
- 17 problems,
  - **Day0:** 3 problems
  - **Day1:** 7 problems
  - **Day2:** 7 problems
- Problems are **in order of difficulty!**
- Leader board will be frozen in **the last 6 hours**.

KU
ACM

Colorful Programming!



Prizes

- **1st Place :** 250₺ Amazon Gift Card
- **2nd Place :** 250₺ Amazon Gift Card
- **3rd Place :** 250₺ Amazon Gift Card

- **4th Place :** 200₺ Amazon Gift Card
- **5th Place :** 200₺ Amazon Gift Card

- Some gifts to random people in the **top 20!**

Rules

- **Coworking** is strictly **prohibited**.
- ICPC scoring rules.
- Searching on the internet is allowed **if**...

You understand the code you submitted!

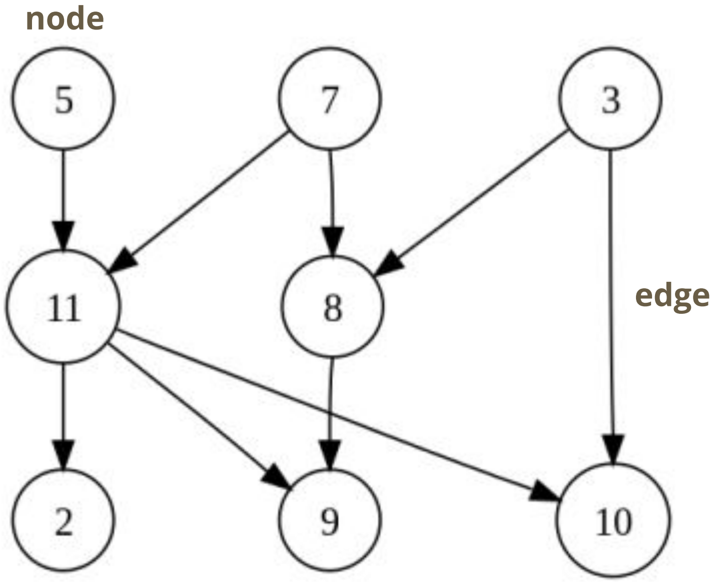
Q & A

KUding Contest

KUding Contest!

Graphs

- A graph consist of
  - Nodes / Vertices
  - Edges
- A graph might be
  - Directed
  - Undirected



This is a directed graph

KU ACM

How to Graph?

- Edge Lists
- Adjacency Matrices
- Adjacency Lists
- OOP

How to Graph?

- **Edge Lists**
- Adjacency Matrices
- Adjacency Lists
- OOP

- Useful for iterating all the edges
- All edges are stored once

- Hard to determine connectivity
- Hard to find edges of a node

How to Graph?

- Edge Lists
- **Adjacency Matrices**
- Adjacency Lists
- OOP

- Easy to check connectivity

- Hard to iterate over all edges of a node

How to Graph?

- Edge Lists
- Adjacency Matrices
- **Adjacency Lists**
- OOP

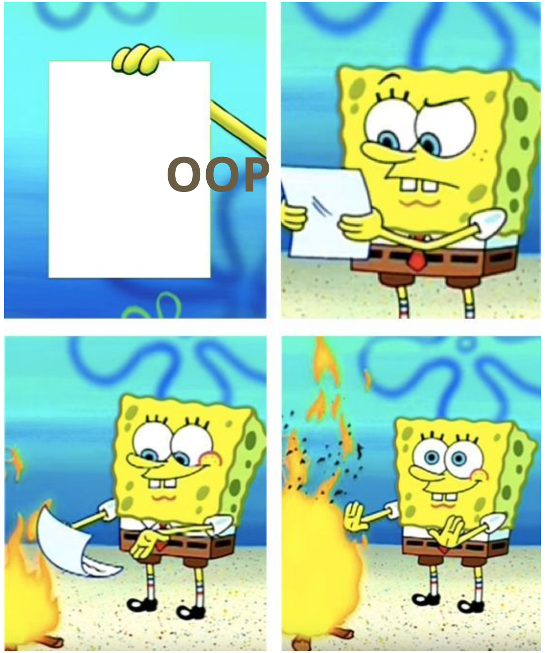
- Efficient memory usage
- Easy to iterate over all adjacents of a node

- Determining connectivity might be hard

How to Graph?

- Edge Lists
- Adjacency Matrices
- Adjacency Lists
- **OOP**

- Not for competitive programming



Traversing The Graph

Usually we need to **traverse** the graph to:

- Examine relations
- Reach elements

Traversing The Graph

Usually we need to **traverse** the graph to:

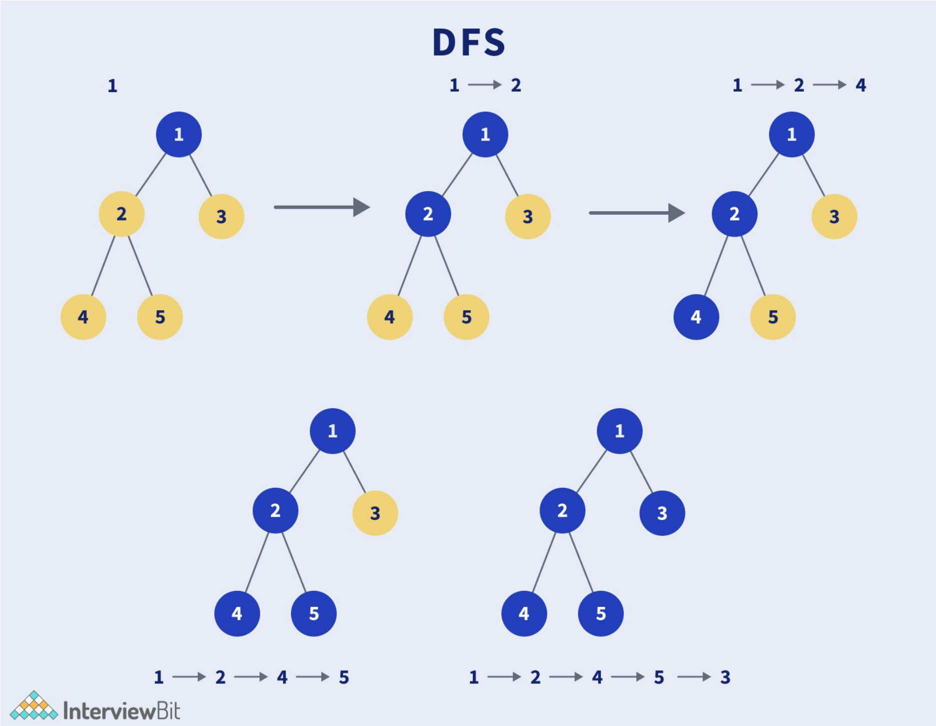
- Examine relations
- Reach elements

Two of the most popular ways of achieving this:

- DFS
- BFS

Depth First Search

- Goal:
  - Visit all the nodes
  - Go ahead if possible
- Can be implemented
  - Recursively
  - Iteratively
- Complexity O(V + E)
  - V is the number of vertices
  - E is the number of edges



KU ACM

Let’s Apply

- **Question 1**: Given an undirected graph, find out whether there is a path between any two pair of nodes.

Let’s Apply

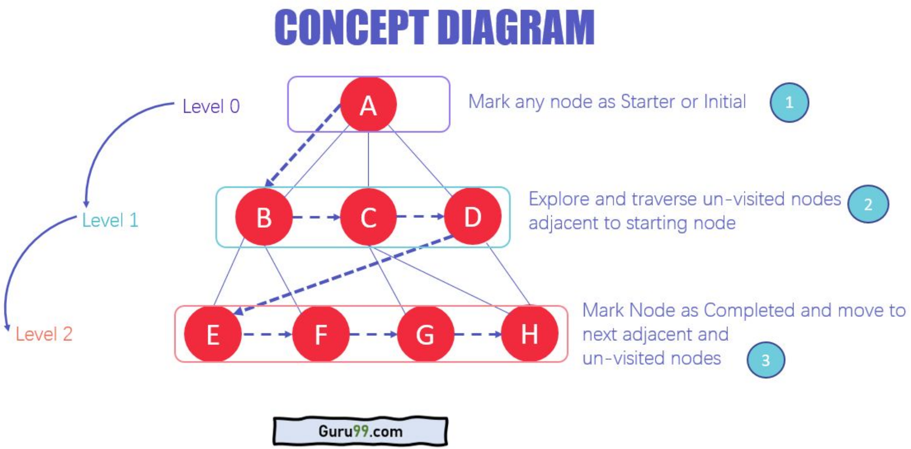
- **Question 1**: Given an undirected graph, find out whether there is a path between any two pair of nodes.

There is a Problem:

How to avoid cycles?

Breadth First Search (BFS)

- Goal:
  - Visit all the nodes
  - Visit all same level nodes
- Layerwise traverse
- Useful for finding shortest path
- Complexity is O(V+E)


CONCEPT DIAGRAM
Level 0
Level 1
Level 2
A
B
C
D
E
F
G
H
Mark any node as Starter or Initial
1
Explore and traverse un-visited nodes adjacent to starting node
2
Mark Node as Completed and move to next adjacent and un-visited nodes
3
Guru99.com

KU ACM

Let's Apply

- **Question 2**: Given a graph, a source and a destination, find the shortest path from source to destination.

Numbers!

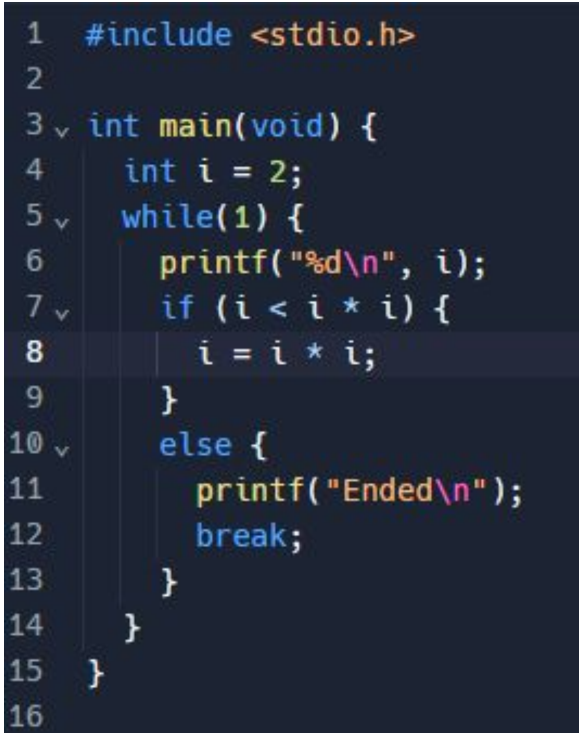
- Can a computer represent **1**?

Numbers!

- Can a computer represent **1**?
- What about **10^9**?

Numbers!

- Can a computer represent **1**?
- What about **10^9**?
- There is **a limit**!



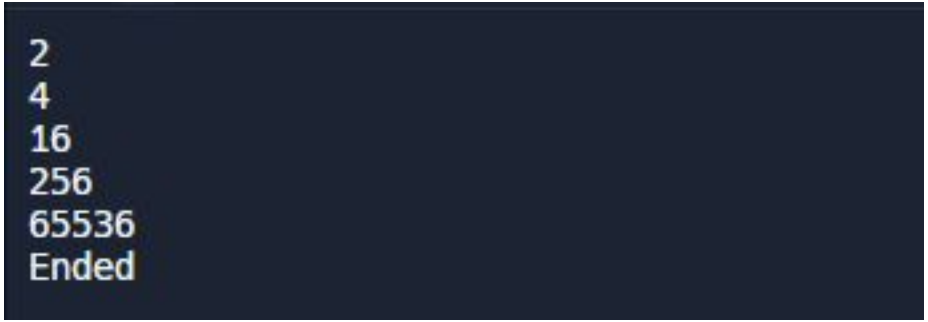
```c
#include <stdio.h>

int main(void) {
  int i = 2;
  while(1) {
    printf("%d\n", i);
    if (i < i * i) {
      i = i * i;
    }
    else {
      printf("Ended\n");
      break;
    }
  }
}

```

Numbers!

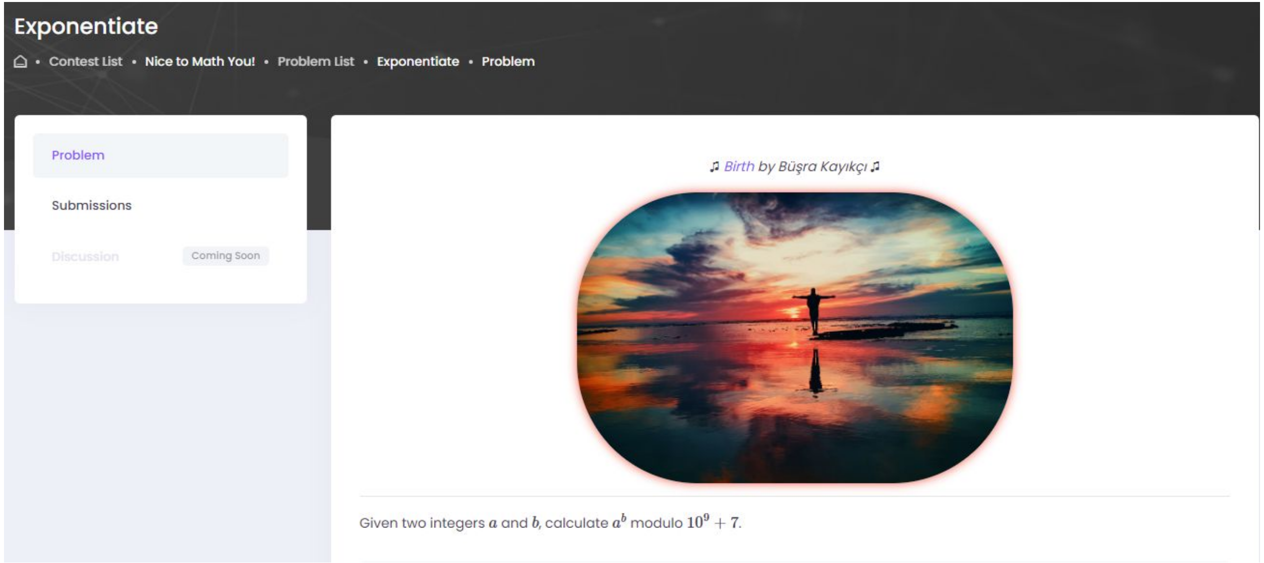
- Can a computer represent **1**?
- What about **10^9**?
- There is **a limit**!
- **Overflow!**



```
2
4
16
256
65536
Ended
```

Let’s Apply

- **Exponentiate**



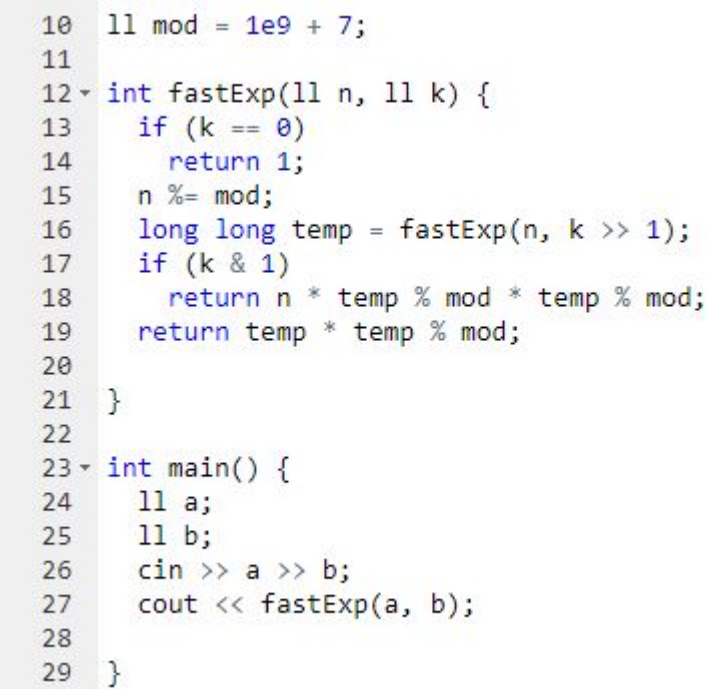
Exponentiate

⌂ • Contest List • Nice to Math You! • Problem List • Exponentiate • Problem

♫ *Birth by Büşra Kayıkçı* ♫

Given two integers a and b, calculate a^b modulo $10^9 + 7$.

What if you code in C++?



```cpp
ll mod = 1e9 + 7;

int fastExp(ll n, ll k) {
  if (k == 0)
    return 1;
  n %= mod;
  long long temp = fastExp(n, k >> 1);
  if (k & 1)
    return n * temp % mod * temp % mod;
  return temp * temp % mod;

}

int main() {
  ll a;
  ll b;
  cin >> a >> b;
  cout << fastExp(a, b);

}
```

End Feedback

Stay with KU ACM!