lcrpmbVX|()[]!@<>:##1
12

Exercise 1: Mutable string

Create a data type that supports the following operations on a string: `get(int i)`, `insert(int i, char c)`, and `delete(int i)`, where `get` returns the ith character of the string, insert inserts the character `c` and makes it the ith character, and **`delete`** deletes the ith character. Use a binary search tree.

Hint: Use a BST (with key = real number between 0 and 1, value = character) so that the inorder traversal of the tree yields the characters in the appropriate order. Use `select()` to find the ith element. When inserting a character at position i, choose the real number to be the average of the keys currently at positions $i-1$ and i.

Use the Red-Black tree implementation from the lectures (`red-black-bst.hpp` attached).

Exercise 2: Graph Operations

In this problem, you are working with an undirected graph represented using an adjacency list. Your task is to implement a series of functions to perform different operations on the graph.

Operations to Implement

1. **Find the Vertex with Maximum Degree:** Write a function that finds and returns the vertex with the maximum degree in the graph. The degree of a vertex is the number of edges connected to it.
2. **Find the Degree of a Vertex:** Write a function that takes a vertex as input and returns the degree (the number of edges) of that vertex.
3. **Find the Vertex with Minimum Degree:** Write a function that finds and returns the vertex with the minimum degree in the graph.

Input:

- An integer n, representing the number of vertices.
- A list of edges, where each edge is a pair of integers $[u, v]$ representing an undirected edge between vertices u and v.
- A list of vertices to check for degree queries.

Output:

- Output the vertex with the highest degree.
- Output the vertex with the lowest degree.
- Output the degree of the specified vertices.

Example:

Input:

```
n = 6
edges = [[0, 1], [0, 2], [3, 4], [4, 5], [2, 4]]
verticesToCheck = [0, 1, 2, 3]
```

Output:

```
Vertex with maximum degree: 4
Vertex with minimum degree: 1
Degree of vertices:
Vertex 0: Degree 2
Vertex 1: Degree 1
Vertex 2: Degree 2
Vertex 3: Degree 1
```