Practical 6

Name: Parth M. Pathak

Section: A6

Batch/Roll Number: B3/47

Task 1

Aim: A university digital library system stores frequently accessed books using a binary search

mechanism. The library admin wants to minimize the average search time for book lookups by

arranging the book IDs optimally in a binary search tree.

Each book ID has a probability of being searched successfully and an associated probability for

unsuccessful searches (when a book ID does not exist between two keys).

Your task is to determine the minimum expected cost of searching using an Optimal Binary

Search Tree (OBST).

Input Format

First line: integer n — number of book IDs.

Second line: n integers representing the sorted book IDs (keys).

Third line: n real numbers — probabilities of successful searches (p[i]).

Fourth line: n+1 real numbers — probabilities of unsuccessful searches (q[i]).

Keys: 10 20 30 40

P[i]: 0.1 0.2 0.4 0.3

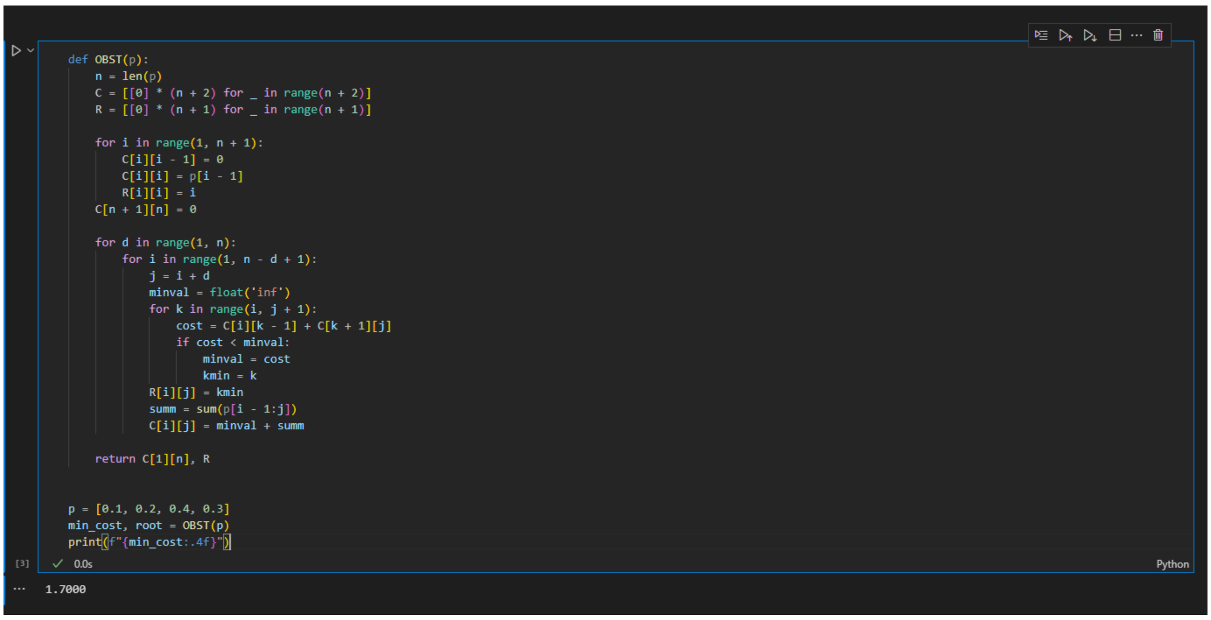
Q[i]: 0.05 0.1 0.05 0.05 0.1

Code:

For Successful:

```python
def OBST(p):
    n = len(p)
    C = [[0] * (n + 2) for _ in range(n + 2)]
    R = [[0] * (n + 1) for _ in range(n + 1)]

    for i in range(1, n + 1):
        C[i][i - 1] = 0
        C[i][i] = p[i - 1]
        R[i][i] = i
    C[n + 1][n] = 0

    for d in range(1, n):
        for i in range(1, n - d + 1):
            j = i + d
            minval = float('inf')
            for k in range(i, j + 1):
                cost = C[i][k - 1] + C[k + 1][j]
                if cost < minval:
                    minval = cost
                    kmin = k
            R[i][j] = kmin
            summ = sum(p[i - 1:j])
            C[i][j] = minval + summ

    return C[1][n], R


p = [0.1, 0.2, 0.4, 0.3]
min_cost, root = OBST(p)
print(f"{min_cost:.4f}")
```

Output:



```python
def OBST(p):
    n = len(p)
    C = [[0] * (n + 2) for _ in range(n + 2)]
    R = [[0] * (n + 1) for _ in range(n + 1)]

    for i in range(1, n + 1):
        C[i][i - 1] = 0
        C[i][i] = p[i - 1]
        R[i][i] = i
    C[n + 1][n] = 0

    for d in range(1, n):
        for i in range(1, n - d + 1):
            j = i + d
            minval = float('inf')
            for k in range(i, j + 1):
                cost = C[i][k - 1] + C[k + 1][j]
                if cost < minval:
                    minval = cost
                    kmin = k
            R[i][j] = kmin
            summ = sum(p[i - 1:j])
            C[i][j] = minval + summ

    return C[1][n], R


p = [0.1, 0.2, 0.4, 0.3]
min_cost, root = OBST(p)
print(f"{min_cost:.4f}")
```

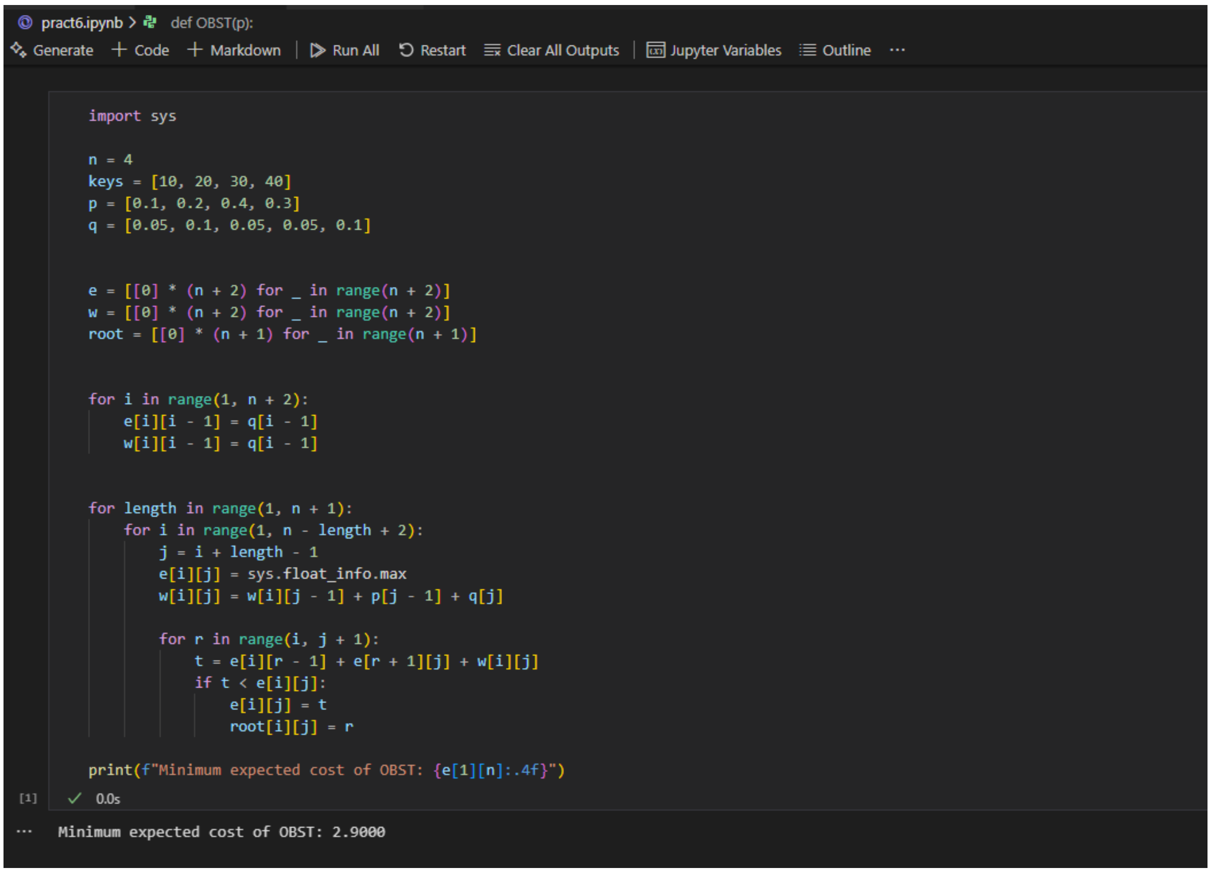
```
1.7000
```

For Successful and Unsuccessful Search:

Code:

```python
import sys

n = 4
keys = [10, 20, 30, 40]
p = [0.1, 0.2, 0.4, 0.3]
q = [0.05, 0.1, 0.05, 0.05, 0.1]


e = [[0] * (n + 2) for _ in range(n + 2)]
w = [[0] * (n + 2) for _ in range(n + 2)]
root = [[0] * (n + 1) for _ in range(n + 1)]


for i in range(1, n + 2):
    e[i][i - 1] = q[i - 1]
    w[i][i - 1] = q[i - 1]


for length in range(1, n + 1):
    for i in range(1, n - length + 2):
        j = i + length - 1
        e[i][j] = sys.float_info.max
        w[i][j] = w[i][j - 1] + p[j - 1] + q[j]

        for r in range(i, j + 1):
            t = e[i][r - 1] + e[r + 1][j] + w[i][j]
            if t < e[i][j]:
                e[i][j] = t
                root[i][j] = r

print(f"Minimum expected cost of OBST: {e[1][n]:.4f}")
```

Output:



```
import sys

n = 4
keys = [10, 20, 30, 40]
p = [0.1, 0.2, 0.4, 0.3]
q = [0.05, 0.1, 0.05, 0.05, 0.1]


e = [[0] * (n + 2) for _ in range(n + 2)]
w = [[0] * (n + 2) for _ in range(n + 2)]
root = [[0] * (n + 1) for _ in range(n + 1)]


for i in range(1, n + 2):
    e[i][i - 1] = q[i - 1]
    w[i][i - 1] = q[i - 1]


for length in range(1, n + 1):
    for i in range(1, n - length + 2):
        j = i + length - 1
        e[i][j] = sys.float_info.max
        w[i][j] = w[i][j - 1] + p[j - 1] + q[j]

        for r in range(i, j + 1):
            t = e[i][r - 1] + e[r + 1][j] + w[i][j]
            if t < e[i][j]:
                e[i][j] = t
                root[i][j] = r

print(f"Minimum expected cost of OBST: {e[1][n]:.4f}")
```

```
Minimum expected cost of OBST: 2.9000
```

Task 2

GFG link: https://www.geeksforgeeks.org/problems/optimal-binary-search-tree2214/1