

Competitive Programming

Assignment-04

2303A51463

B-07

WEEK7_4: Practical Exercises with Fenwick Trees -Binary Indexed Trees

Problem: Library Book Borrowing Records:

Algorithm:

1. Read integer **T** (number of test cases).
2. For each test case:
 - a. Read integer **N** (number of days).
 - b. Read array **A[1...N]** representing books borrowed each day.
 - c. Initialize a Fenwick Tree **BIT[1...N]** with all values as 0.
 - d. Build the Fenwick Tree:
 - i. For $i = 1$ to N , perform $\text{update}(i, A[i])$.
 - e. Read integer **Q** (number of queries).
 - f. For each query:
 - i. If query is **SUM x**:
 1. Compute prefix sum from Day 1 to Day x using Fenwick Tree.
 2. Print the result.
 - ii. If query is **UPDATE i val**:
 1. Increase books borrowed on Day i by val .
 2. Perform $\text{update}(i+1, val)$ in Fenwick Tree.
3. End.

Code :

```
ass4.4.py > ...
1  class FenwickTree:
2      def __init__(self,n):
3          self.n=n
4          self.bit=[0]*(n+1)
5      def update(self,i,val):
6          while i<=self.n:
7              self.bit[i]+=val
8              i+=i&-i
9      def query(self,i):
10         s=0
11         while i>0:
12             s+=self.bit[i]
13             i-=i&-i
14         return s
15
16 T=int(input())
17 for _ in range(T):
18     N=int(input())
19     arr=list(map(int,input().split()))
20     ft=FenwickTree(N)
21     for i in range(N):
22         ft.update(i+1,arr[i])
23     Q=int(input())
24     for _ in range(Q):
25         q=input().split()
26         if q[0]=="SUM":
27             x=int(q[1])
28             print(ft.query(x))
29         else:
30             i=int(q[1])+1
31             val=int(q[2])
32             ft.update(i,val)
```

Output:

```
● PS C:\Users\harsh\Desktop\CP> & C:/Users/harsh/AppData/Local/Programs/Python/Python37-32/ass4.4.py
1
6
12 15 10 20 18 25
4
SUM 4
57
UPDATE 3 5
SUM 4
62
SUM 6
105
```

Problem: Daily Water Consumption Analysis Using Fenwick Tree

Algorithm:

1. Read integer **N** (number of days).
2. Read array **A[1...N]** representing daily water consumption.
3. Initialize a Fenwick Tree **BIT[1...N]** and build it using the array values.
4. Read integer **Q** (number of operations).
5. For each operation:
 - If operation is **QUERY d**, print total water consumption from Day 1 to Day **d**.
 - If operation is **UPDATE i x**, update water consumption on Day **i** and modify Fenwick Tree.
6. End.

```
⌚ ass4.4.2.py > ...
1  class FenwickTree:
2      def __init__(self,n):
3          self.n=n
4          self.bit=[0]*(n+1)
5      def update(self,i,val):
6          while i<=self.n:
7              self.bit[i]+=val
8              i+=i&-i
9      def query(self,i):
10         s=0
11         while i>0:
12             s+=self.bit[i]
13             i-=i&-i
14         return s
15
16 N=int(input())
17 arr=list(map(int,input().split()))
18 ft=FenwickTree(N)
19 for i in range(N):
20     ft.update(i+1,arr[i])
21 Q=int(input())
22 for _ in range(Q):
23     op=input().split()
24     if op[0]=="QUERY":
25         d=int(op[1])
26         print(ft.query(d))
27     else:
28         i=int(op[1])
29         x=int(op[2])
30         delta=x-arr[i-1]
31         arr[i-1]=x
32         ft.update(i,delta)
```

Output:

```
● PS C:\Users\harsh\Desktop\CP> & C:/Users/harsh/AppData/Roaming/Python/3.8/site-packages/ass4.4.2.py
6
120 135 110 150 140 125
3
QUERY 4
515
UPDATE 2 145
QUERY 4
525
```

WEEK7_5_Practical Exercises with Fenwick Trees –Binary Indexed Trees

Problem: Monthly Electricity Consumption Tracking

Algorithm:

1. Read integer **T** (number of test cases).
2. For each test case:
 - a. Read integer **N** (number of days).
 - b. Read array **A[1...N]** representing daily electricity consumption.
3. Initialize a Fenwick Tree **BIT[1...N]** with all values as 0.
4. Build the Fenwick Tree by performing **update(i, A[i])** for **i = 1 to N**.
5. Read integer **Q** (number of operations).
6. For each operation:
 - If operation is **SUM x**, compute and print prefix sum from Day 1 to Day **x**.
 - If operation is **UPDATE i val**, increase the value on Day **i** by **val** and update Fenwick Tree.
7. End.

```

ass4.5.py > ...
1  class FenwickTree:
2      def __init__(self,n):
3          self.n=n
4          self.bit=[0]^(n+1)
5      def update(self,i,val):
6          while i<=self.n:
7              self.bit[i]+=val
8              i+=i&-i
9      def query(self,i):
10         s=0
11         while i>0:
12             s+=self.bit[i]
13             i-=i&-i
14         return s
15
16 T=int(input())
17 for _ in range(T):
18     N=int(input())
19     arr=list(map(int,input().split()))
20     ft=FenwickTree(N)
21     for i in range(N):
22         ft.update(i+1,arr[i])
23
24     Q=int(input())
25     for _ in range(Q):
26         op=input().split()
27         if op[0]=="SUM":
28             x=int(op[1])
29             print(ft.query(x))
30         else:
31             i=int(op[1])
32             val=int(op[2])
33             ft.update(i,val)

```

Output:

```

● PS C:\Users\harsh\Desktop\CP> & C:/Users/harsh/AppDat
P/ass4.5.py
1
7
30 28 35 40 33 38 36
3
SUM 5
166
UPDATE 4 -3
SUM 5
163

```

Problem: Daily Mobile Data Usage Analysis Using Fenwick Tree

Algorithm:

- 1.** Read integer **N** (number of days).
- 2.** Read array **A[1...N]** representing daily mobile data usage in MB.
- 3.** Initialize a Fenwick Tree **BIT[1...N]** with all values as 0.
- 4.** Build the Fenwick Tree by performing $\text{update}(i, A[i])$ for $i = 1$ to N .
- 5.** Read integer **Q** (number of operations).
- 6.** For each operation:
 - If operation is **QUERY d**, compute and print total mobile data usage from Day 1 to Day d .
 - If operation is **UPDATE i x**,
 - Compute $\text{delta} = x - A[i]$
 - Update $A[i] = x$
 - Perform $\text{update}(i, \text{delta})$ in Fenwick Tree.
- 7.** End.

```

ass.4.5.2.py > ...
1  class FenwickTree:
2      def __init__(self,n):
3          self.n=n
4          self.bit=[0]^(n+1)
5      def update(self,i,val):
6          while i<=self.n:
7              self.bit[i]+=val
8              i+=i&-i
9
10     def query(self,i):
11         s=0
12         while i>0:
13             s+=self.bit[i]
14             i-=i&-i
15         return s
16
17 N=int(input())
18 arr=list(map(int,input().split()))
19 ft=FenwickTree(N)
20 for i in range(N):
21     ft.update(i+1,arr[i])
22 Q=int(input())
23 for _ in range(Q):
24     op=input().split()
25     if op[0]=="QUERY":
26         d=int(op[1])
27         print(ft.query(d))
28     else:
29         i=int(op[1])
30         x=int(op[2])
31         delta=x-arr[i-1]
32         arr[i-1]=x
33         ft.update(i,delta)

```

Output:

```

PS C:\Users\harsh\Desktop\CP> & C:/Users/harsh/AppData/
P/ass.4.5.2.py
● 7
500 650 400 800 550 700 600
3
QUERY 6
3600
UPDATE 3 480
QUERY 6
3680

```

