

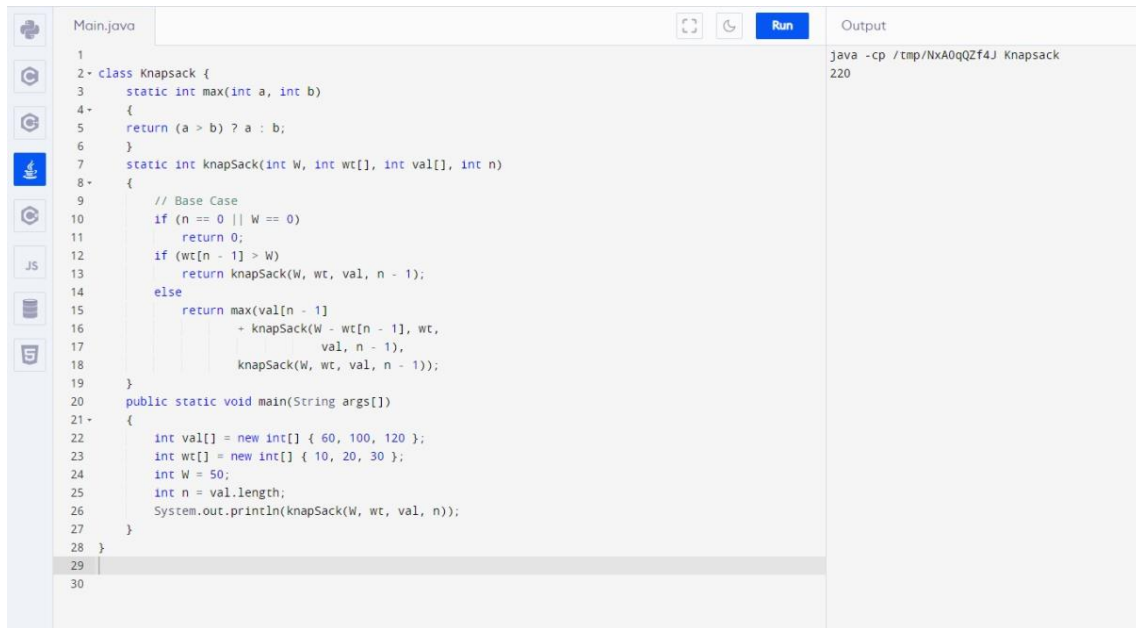
## Week 4

***Name- Akanksha Yadav***

**Roll no- 1900290120008**

### 1. 0/1 knapsack

```
class Knapsack {
    static int max(int a, int b)
    {
        return (a > b) ? a : b;
    }
    static int knapSack(int W, int wt[], int val[], int n)
    {
        // Base Case
        if (n == 0 || W == 0)
            return 0;
        if (wt[n - 1] > W)
            return knapSack(W, wt, val, n - 1);
        else
            return max(val[n - 1]
                        + knapSack(W - wt[n - 1], wt,
                                   val, n - 1),
                        knapSack(W, wt, val, n - 1));
    }
    public static void main(String args[])
    {
        int val[] = new int[] { 60, 100, 120 };
        int wt[] = new int[] { 10, 20, 30 };
        int W = 50;
        int n = val.length;
        System.out.println(knapSack(W, wt, val, n));
    }
}
```



```
1
2 class Knapsack {
3     static int max(int a, int b)
4     {
5         return (a > b) ? a : b;
6     }
7     static int knapSack(int W, int wt[], int val[], int n)
8     {
9         // Base Case
10        if (n == 0 || W == 0)
11            return 0;
12        if (wt[n - 1] > W)
13            return knapSack(W, wt, val, n - 1);
14        else
15            return max(val[n - 1]
16                      + knapSack(W - wt[n - 1], wt,
17                                val, n - 1),
18                      knapSack(W, wt, val, n - 1));
19    }
20    public static void main(String args[])
21    {
22        int val[] = new int[] { 60, 100, 120 };
23        int wt[] = new int[] { 10, 20, 30 };
24        int W = 50;
25        int n = val.length;
26        System.out.println(knapSack(W, wt, val, n));
27    }
28 }
29
30
```

Output

```
java -cp /tmp/NxA0qQZf4J Knapsack
220
```

## 2. LCS algo

```
class LCS_ALGO {
    static void lcs(String S1, String S2, int m, int n) {
        int[][] LCS_table = new int[m + 1][n + 1];
        for (int i = 0; i <= m; i++) {
            for (int j = 0; j <= n; j++) {
                if (i == 0 || j == 0)
                    LCS_table[i][j] = 0;
                else if (S1.charAt(i - 1) == S2.charAt(j - 1))
                    LCS_table[i][j] = LCS_table[i - 1][j - 1] + 1;
                else
                    LCS_table[i][j] = Math.max(LCS_table[i - 1][j], LCS_table[i][j - 1]);
            }
        }
    }
}
```

```
int index = LCS_table[m][n];
```

```
int temp = index;
```

```
char[] lcs = new char[index + 1];
```

```
lcs[index] = '\0';
```

```
int i = m, j = n;
```

```
while (i > 0 && j > 0) {
```

```
    if (S1.charAt(i - 1) == S2.charAt(j - 1)) {
```

```
        lcs[index - 1] = S1.charAt(i - 1);
```

```

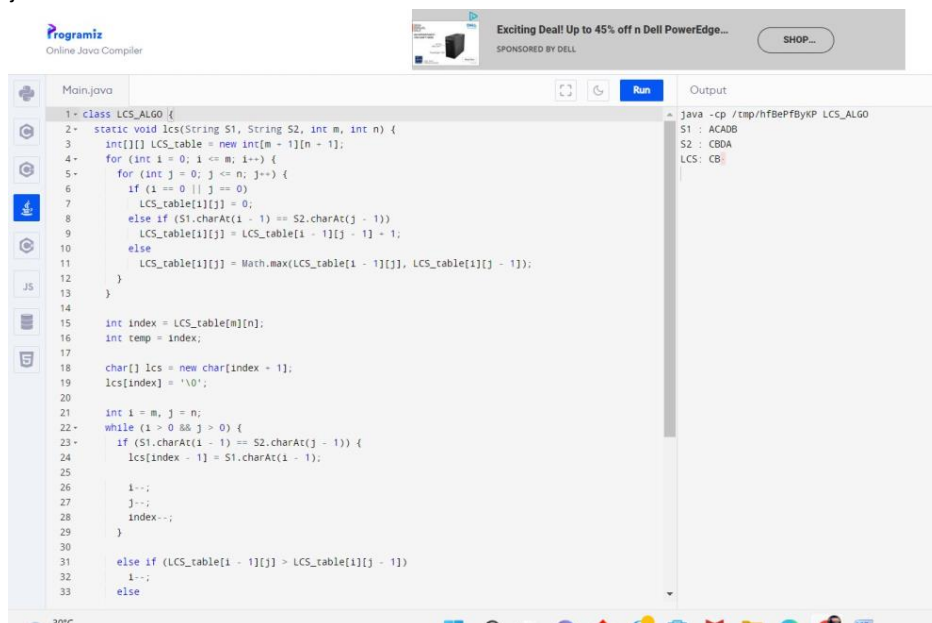
        i--;
        j--;
        index--;
    }

    else if (LCS_table[i - 1][j] > LCS_table[i][j - 1])
        i--;
    else
        j--;
    }

    System.out.print("S1 : " + S1 + "\nS2 : " + S2 + "\nLCS: ");
    for (int k = 0; k <= temp; k++)
        System.out.print(lcs[k]);
    System.out.println("");
}

public static void main(String[] args) {
    String S1 = "ACADB";
    String S2 = "CBDA";
    int m = S1.length();
    int n = S2.length();
    lcs(S1, S2, m, n);
}
}

```



The screenshot shows an online Java compiler interface. At the top, there's a banner for Dell PowerEdge with a 45% off deal. Below the banner, the code editor shows the same Java code as in the previous block. The output window on the right displays the results of the program execution:

```

S1 : ACADB
S2 : CBDA
LCS: CB

```

```

29     }
30
31     else if (LCS_table[i - 1][j] > LCS_table[i][j - 1])
32         i--;
33     else
34         j--;
35 }
36 System.out.print("S1 : " + S1 + "\nS2 : " + S2 + "\nLCS: ");
37 for (int k = 0; k <= temp; k++)
38     System.out.print(lcs[k]);
39 System.out.println("");
40 }
41
42 public static void main(String[] args) {
43     String S1 = "ACADB";
44     String S2 = "CBDA";
45     int m = S1.length();
46     int n = S2.length();
47     lcs(S1, S2, m, n);
48 }
49 }

```

### 3. MCM algo

```

class MatrixChainMultiplication {
    static int MatrixChainOrder(int p[], int i, int j)
    {
        if (i == j)
            return 0;


        int min = Integer.MAX_VALUE;
        for (int k = i; k < j; k++)
        {
            int count = MatrixChainOrder(p, i, k)
                          + MatrixChainOrder(p, k + 1, j)
                          + p[i - 1] * p[k] * p[j];

            if (count < min)
                min = count;
        }
        return min;
    }

    public static void main(String args[])
    {
        int arr[] = new int[] { 1, 2, 3, 4, 3 };
        int n = arr.length;

        System.out.println(
            + MatrixChainOrder(arr, 1, n - 1));
    }
}

```



Online Java Compiler

Exciting Deal! Up to 45% off n Dell PowerEdge...

SPONSORED BY DELL

SHOP...

Main.java

```

1 class MatrixChainMultiplication {
2     static int MatrixChainOrder(int p[], int i, int j)
3     {
4         if (i == j)
5             return 0;
6
7         int min = Integer.MAX_VALUE;
8         for (int k = i; k < j; k++)
9         {
10            int count = MatrixChainOrder(p, i, k)
11                      + MatrixChainOrder(p, k + 1, j)
12                      + p[i - 1] * p[k] * p[j];
13
14            if (count < min)
15                min = count;
16        }
17        return min;
18    }
19    public static void main(String args[])
20    {
21        int arr[] = new int[] { 1, 2, 3, 4, 3 };
22        int n = arr.length;
23
24        System.out.println(
25            MatrixChainOrder(arr, 1, n - 1));
26    }
27 }
28

```


Output

```

java -cp /tmp/nDo5o06TVh MatrixChainMultiplicatio
Minimum number of multiplications is 30

```

30°C Cloudy



#### 4. Unbounded knapsack

```

class Knapsack {
    static int max(int a, int b) { return (a > b) ? a : b; }
    static int unboundedKnapsack(int W, int wt[], int val[],
                                int idx)
    {
        if (idx == 0) {
            return (W / wt[0]) * val[0];
        }
        int notTake
            = 0 + unboundedKnapsack(W, wt, val, idx - 1);
        int take = Integer.MIN_VALUE;
        if (wt[idx] <= W) {
            take = val[idx]
                + unboundedKnapsack(W - wt[idx], wt, val,
                                    idx);
        }
        return max(take, notTake);
    }
    public static void main(String args[])
    {
        int W = 100;
        int val[] = { 10, 30, 20 };
        int wt[] = { 5, 10, 15 };
    }
}

```

```

    int n = val.length;
    System.out.println(
        unboundedKnapsack(W, wt, val, n - 1));
}
}

```

The screenshot shows the Programiz Online Java Compiler interface. The code editor contains the following Java code:

```

1- class Knapsack {
2-     static int max(int a, int b) { return (a > b) ? a : b; }
3-     static int unboundedKnapsack(int W, int wt[], int val[],
4-                                 int idx)
5-     {
6-         if (idx == 0) {
7-             return (W / wt[0]) * val[0];
8-         }
9-         int notTake
10-            = 0 + unboundedKnapsack(W, wt, val, idx - 1);
11-         int take = Integer.MIN_VALUE;
12-         if (wt[idx] <= W) {
13-             take = val[idx]
14-                 + unboundedKnapsack(W - wt[idx], wt, val,
15-                                     idx);
16-         }
17-         return max(take, notTake);
18-     }
19-     public static void main(String args[])
20-     {
21-         int W = 100;
22-         int val[] = { 10, 30, 20 };
23-         int wt[] = { 5, 10, 15 };
24-         int n = val.length;
25-         System.out.println(
26-             unboundedKnapsack(W, wt, val, n - 1));
27-     }
28- }
29-
30-

```

The output window shows the command `java -cp /tmp/SQmldj3nEd Knapsack` and the result `300`.

## 5. LIS Algo

```


class LIS {
    static int max_ref;
    static int _lis(int arr[], int n)
    {
        if (n == 1)
            return 1;
        int res, max_ending_here = 1;
        for (int i = 1; i < n; i++) {
            res = _lis(arr, i);
            if (arr[i - 1] < arr[n - 1]
                && res + 1 > max_ending_here)
                max_ending_here = res + 1;
        }
        if (max_ref < max_ending_here)
            max_ref = max_ending_here;
        return max_ending_here;
    }
    static int lis(int arr[], int n)
    {

```


```

        max_ref = 1;
        _lis(arr, n);
        return max_ref;
    }
    public static void main(String args[])
    {
        int arr[] = { 10, 22, 9, 33, 21, 50, 41, 60 };
        int n = arr.length;
        System.out.println("Length of lis is " + lis(arr, n)
                            + "\n");
    }
}


```




Programiz  
Online Java Compiler



**LOOKING TO LEARN PROGRAMMING?**  
Start your programming journey with Programiz **AT NO COST.**



Main.java	Output
<pre> 1- class LIS { 2     static int max_ref; 3     static int _lis(int arr[], int n) 4+ { 5         if (n == 1) 6             return 1; 7         int res, max_ending_here = 1; 8+         for (int i = 1; i &lt; n; i++) { 9             res = _lis(arr, i); 10            if (arr[i - 1] &lt; arr[n - 1]) 11                &amp;&amp; res + 1 &gt; max_ending_here) 12                max_ending_here = res + 1; 13        } 14        if (max_ref &lt; max_ending_here) 15            max_ref = max_ending_here; 16        return max_ending_here; 17    } 18    static int lis(int arr[], int n) 19+ { 20        max_ref = 1; 21        _lis(arr, n); 22        return max_ref; 23    } 24    public static void main(String args[]) 25+ { 26        int arr[] = { 10, 22, 9, 33, 21, 50, 41, 60 }; 27        int n = arr.length; 28        System.out.println("Length of lis is " + lis(arr, n) 29                            + "\n"); 30    } 31 } 32 </pre>	<pre> java -cp /tmp/5Qmldj3nEd LIS Length of lis is 5 </pre>



30°C  
Cloudy

