**A1 FIBONACCI SERIES USING ITERATIVE APPROACH:**

import java.util.Scanner;

public class FiboIterative {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

int a = 0;

int b = 1;

System.out.println(a);

System.out.println(b);

for(int i=2; i<n; i++){

int c = a + b;

System.out.println(c);

a=b;

b=c;

}

}

}

**A1 FIBONACCI SERIES USING RECURSIVE APPROACH:**

import java.util.Scanner;

public class FiboRecursive {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

for(int i=0; i<n; i++){

System.out.println(fibo(i));

}

}

static int fibo(int n)

{

if(n<=1){

return n;

}

else{

return fibo(n-1) + fibo(n-2);

}

}

}

**A3 FRACTIONAL KNAPSACK PROBLEM**

import java.io.IOException;

import java.util.Scanner;

public class KnapsackProblem {

public static void main(String args[]) throws IOException

{

int i,j=0,max\_qty,m,n;

float sum=0,max;

Scanner sc = new Scanner(System.in);

int array[][]=new int[2][20];

System.out.println("Enter no of items");

n=sc.nextInt();

System.out.println("Enter the weights of each items");

for(i=0;i<n;i++)

array[0][i]=sc.nextInt();

System.out.println("Enter the values of each items");

for(i=0;i<n;i++)

array[1][i]=sc.nextInt();

System.out.println("Enter maximum volume of knapsack :");

max\_qty=sc.nextInt();

m=max\_qty;

while(m>=0)

{

max=0;

for(i=0;i<n;i++)

{

if(((float)array[1][i])/((float)array[0][i])>max)

{

max=((float)array[1][i])/((float)array[0][i]);

j=i;

}

}

if(array[0][j]>m)

{

System.out.println("Quantity of item number: " + (j+1) + " added is " +m);

sum+=m\*max;

m=-1;

}

else

{

System.out.println("Quantity of item number: " + (j+1) + " added is " + array[0][j]);

m-=array[0][j];

sum+=(float)array[1][j];

array[1][j]=0;

}

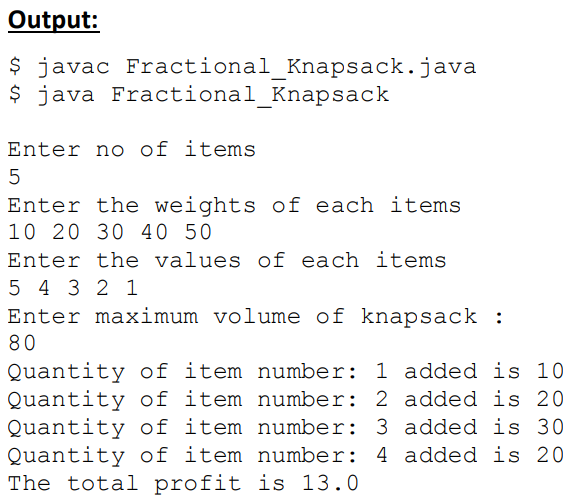
}

System.out.println("The total profit is " + sum);

sc.close();

}

}



**0/1 KNAPSACK PROBLEM**

import java.util.Scanner;

public class Zero\_One\_Knapsack

{

public void solve(int[] wt, int[] val, int W, int N)

{

int NEGATIVE\_INFINITY = Integer.MIN\_VALUE;

int[][] m = new int[N + 1][W + 1];

int[][] sol = new int[N + 1][W + 1];

for (int i = 1; i <= N; i++)

{

for (int j = 0; j <= W; j++)

{

int m1 = m[i - 1][j];

int m2 = NEGATIVE\_INFINITY;

if (j >= wt[i])

m2 = m[i - 1][j - wt[i]] + val[i];

m[i][j] = Math.max(m1, m2);

sol[i][j] = m2 > m1 ? 1 : 0;

}

}

int[] selected = new int[N + 1];

for (int n = N, w = W; n > 0; n--)

{

if (sol[n][w] != 0)

{

selected[n] = 1;

w = w - wt[n];

}

else

selected[n] = 0;

}

System.out.print("\nItems with weight ");

for (int i = 1; i < N + 1; i++)

if (selected[i] == 1)

System.out.print(val[i] +" ");

System.out.println("are selected by knapsack algorithm.");

}

public static void main (String[] args)

{

Scanner scan = new Scanner(System.in);

Zero\_One\_Knapsack ks = new Zero\_One\_Knapsack();

System.out.println("Enter number of elements ");

int n = scan.nextInt();

int[] wt = new int[n + 1];

int[] val = new int[n + 1];

System.out.println("Enter weight for "+ n +" elements");

for (int i = 1; i <= n; i++)

wt[i] = scan.nextInt();

System.out.println("Enter value for "+ n +" elements");

for (int i = 1; i <= n; i++)

val[i] = scan.nextInt();

System.out.println("Enter knapsack weight ");

int W = scan.nextInt();

ks.solve(wt, val, W, n);

scan.close();

}

}

