```
public class MainClass {

    public static void main(String args[]){
        Combinations comb = new Combinations();
        System.out.println("F(8,2) is "+comb.sum());
        System.out.println("G(6) is the maximum of "+comb.sixfoldsum1()+" and
"+comb.sixfoldsum2());
    }

}

public class Operation {

    public double rootchoose(int b, int d, int n, int r){
        if(b>d)
            return 1;
        long a = r*choose(n-b, n-d);
        double c = a;
        double f = d;
        double e = (double) 1/f;
        return Math.pow(c, e);
    }

    public long choose(int a, int b){
        if(a<=b)
            return 1;
        long result = 1;
        if(2*b>=a)
            result = factorial(a, b+1)/factorial(a-b,1);
        else
            result = factorial(a, a-b+1)/factorial(b,1);
        return result;
    }

    public long factorial(int a, int b){//multipliction of all numbers a>= c >= b
        if (a<b)
            return 1;
        long result = 1;
        for(int i=b; i<=a; i++)
            result = result*i;
        return result;
    }

}

public class Combinations {
```

```
    Operation op;

    public Combinations(){
        op = new Operation();
    }

    /*
    To use the following code, erase the comment symbol up to the desired dimension.
    Set n and r accordingly.
    */

    public double sum(){
        double maxsum = 0;
        double sum;
        int n = 8;
        int[] b = new int[n+1];
        int r = 2;
        b[n]=n;

        /*we sum among all dimensions from d=1 to d=n-1. We let and b[d] the
corresponding b.
        Therefore, the index of b is the dimension */

        //for(int b16=8; (b16<=17) && (b16<=16); b16++)
        //for(int b15=0; (b15<=b16) && (b15<=15); b15++)
        //for(int b14=0; (b14<=b15) && (b14<=14); b14++)
        //for(int b13=0; (b13<=b14) && (b13<=13); b13++)
        //for(int b12=0; (b12<=13) && (b12<=12); b12++)
        //for(int b11=0; (b11<=b12) && (b11<=11); b11++)
        //for(int b10=0;(b10<=b11) && (b10<=10); b10++)
        //for(int b9=0; (b9<= b10) && (b9<=9); b9++)
        //for(int b8=0; (b8<= b9) && (b8<=8); b8++)
        for(int b7=0; (b7<= 8) && (b7<=7); b7++)
        for(int b6=0; (b6<= b7) && (b6<=6); b6++)
        for(int b5=0; (b5<= b6) && (b5<=5); b5++)
        for(int b4=0; (b4<= b5) && (b4<=4); b4++)
        for(int b3=0; (b3<= b4) && (b3<=3); b3++)
        for(int b2=0; (b2<= b3) && (b2<=2); b2++)
        for(int b1=0; (b1<= b2) && (b1<=1); b1++){
            b[1]=b1;
            b[2]=b2;
            b[3]=b3;
            b[4]=b4;
            b[5]=b5;
            b[6]=b6;
            b[7]=b7;
            //b[8]=b8;
```

```
        //b[9]=b9;
        //b[10]=b10;
        //b[11]=b11;
        //b[12]=b12;
        //b[13]=b13;
        //b[14]=b14;
        //b[15]=b15;
        //b[16]=b16;
        sum=this.sum(b, n, r);
        if(sum>=maxsum)
            maxsum = sum;
    }
    return maxsum;
}

public double sum(int[] b, int n, int r){
    double result = 0;
    for(int i=n; i>0; i--)
        result = result+(b[i]-b[i-1])*op.rootchoose(b[i], i, n, r);
    return result;
}

public double sixfoldsum1(){
    double maxsum = 0;
    double sum;
    for(int b4=0; (b4<= 5) && (b4<=4); b4++)
    for(int b3=0; (b3<= b4) && (b3<=3); b3++)
    for(int b2=0; (b2<= b3) && (b2<=2); b2++)
    for(int b1=0; (b1<= b2) && (b1<=1); b1++){
        int m=0;
        if(b2!=0)
            m=2/b2;
        sum=(6-b4)*op.rootchoose(6, 6, 6, 1)+
                (b4-b3)*op.rootchoose(b4, 4, 6, 1)+
                (b3-b2)*op.rootchoose(b3, 3, 6, 1)+
                (b2-b1)*Math.pow((double) m, 0.5)+
                (b1)*op.rootchoose(b1, 1, 6, 1);
        if(sum>=maxsum)
            maxsum = sum;
    }
    return maxsum;
}

public double sixfoldsum2(){
    double maxsum = 0;
    double sum;
    for(int b4=0; (b4<= 5) && (b4<=4); b4++)
```

```
        for(int b3=0; (b3<= b4) && (b3<=3); b3++)
        for(int m=1; m<=5; m++){
           double b2=(double) 2/m;
           sum=(6-b4)*op.rootchoose(6, 6, 6, 1)+
                  (b4-b3)*op.rootchoose(b4, 4, 6, 1)+
                  (b3-b2)*op.rootchoose(b3, 3, 6, 1)+
                  (b2)*Math.pow((double) m, 0.5);
           if(sum>=maxsum)
              maxsum = sum;
        }
        return maxsum;
     }

}
```

OUTPUT

```
F(8,2) is 14.073117122562193
G(6) is the maximum of 7.786582715936868 and 7.8192360309240305
```