**package** comp352;

**import** java.util.Scanner;

//Yifan Yang 40038814

//COMP 352 assignment 1

//the obbnacci of binary version

**public** **class** BinaryOddStack

{

**private** **static** **int** *numOfCalls* = 0; //record number of this method is called

**public** **static** **int** binaryOdd(**int** k)

{

**int** OddVal;//the value of kth oddnacci number

*numOfCalls*++;

**if**(k == 1 || k == 2|| k==3)//initial

{

OddVal = 1;

}

**else**

{

OddVal = *binaryOdd*(k - 1) + *binaryOdd*(k - 2)+*binaryOdd*(k-3);

}

**return** OddVal;

}

**public** **static** **void** main(String[] args)

{

**long** startTime = System.*currentTimeMillis*();//record starttime

**int** n;

Scanner kb = **new** Scanner(System.***in***);

System.***out***.print("Please enter the a non-negative value to find its Oddnacci sequence: ");

n = kb.nextInt();

**int** OddVal = *binaryOdd*(n);

System.***out***.println("The Oddnacci value for n=" + n + " is:" + OddVal +".");

System.***out***.println("The method has been called " + *numOfCalls* + " times.");

kb.close();

**long** endTime = System.*currentTimeMillis*();//record endtime

System.***out***.println("Running time is " + (endTime - startTime) + " ms");

}

}

**package** comp352;

**import** java.util.Scanner;

//Yifan Yang 40038814

//COMP 352 assignment 1

//the obbnacci of linear version

**public** **class** LinearOddStack

{

**private** **static** **int** *numOfCalls* = 0; //record number of this method is called

**private** **static** **int** *Oddnumber*; //record the number of kth oddnacci

**public** **static** **int**[] linearOdd(**int** k)

{

*numOfCalls*++;

**int**[] A = **new** **int**[3]; //an array A[0] is kth,A[1] is k-1,A[2] is k-2

**int** i = 0, j = 0, m=0;

**if** (k == 1) //(base)

{

i=1;

m=1;

j=1;

A[0] = i; A[1] = j;A[2]=m;

*Oddnumber*=1;

**return** (A); // this will return (1,1,1)

}

**else** **if**( k==2 || k==3){

A = *linearOdd*(k - 1);

i=1;

m=1;

j=1;

A[0] = i; A[1] = j;A[2]=m;

*Oddnumber*=1;

**return** (A); // this will return (1,1,1)

}

**else**

{

A = *linearOdd*(k - 1); //recursion to get kn-1,kn-2,kn-3

i = A[0];

j = A[1];

m = A[2];

A[0] = i + j +m;

A[1] = i;

A[2] = j;

*Oddnumber*=i+j+m;

**return** (A); // this will return (i+j+m, i,j)

}

}

**public** **static** **void** main(String[] args)

{

**long** startTime = System.*currentTimeMillis*();//record starttime

**int** n;

Scanner kb = **new** Scanner(System.***in***);

System.***out***.print("Please enter the a non-negative value to find its Oddnacci sequence: ");

n = kb.nextInt();

*linearOdd*(n);

System.***out***.println("The Oddnacci value for n=" + n + " is:" + *Oddnumber* +".");

System.***out***.println("The method has been called " + *numOfCalls* + " times.");

kb.close();

**long** endTime = System.*currentTimeMillis*(); //record endtime

System.***out***.println("Running time is " + (endTime - startTime) + " ms");

}

}