//Fuzail Khan (Question 1):

#include<stdio.h>  
#include<stdlib.h>  
#include<sys/wait.h>  
#include<unistd.h>

int main(int x, char \*\*y){  
   int n;  
    
   if(x==1){   
        exit(0);  
    }  
    
    n = atoi(y[1]);  
    
    if(n<= 0){  
       printf("\nInsert a positive integer.\n\n");  
        exit(0);  
    }  
    
    if(fork() == 0){   
       while(n > 1){  
           printf("%d ", n);  
           if(n %2 == 0)  
               n = n / 2;  
           else  
               n = 3 \* n + 1;        
       }  
       printf("1\n\n");  
    }  
    else{   
       wait(NULL);  
    }  
    
    return 0;  
}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Hannah Ajayi

Question #2

**import** java.util.Scanner;

**class** PrimeNumber

{

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

{

**int** n,p;

Scanner s=**new** Scanner(System.in);

System.out.println(“Enter number : ”);

n=s.nextInt();

**for**(**int** i=2;i<n;i++)

{

p=0;

**for**(**int** j=2;j<i;j++)

{

**if**(i%j==0)

p=1;

}

**if**(p==0){

System.out.println(i);

}

}

}

}

//Mohit Khan

//Question# 3

import java.util.Scanner;

public class MultiThread {

public static void main(String []args){

int n;

int[] arr;

final int[] minmax = new int[2];

final double[] avgs = new double[1];

Scanner s = new Scanner(System.in);

System.out.print("Enter the number of elements :");

n = s.nextInt();

arr = new int[n];

System.out.println("Enter the elements :");

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

arr[i] = s.nextInt();

Thread avgThread = new Thread(new Runnable() {

@Override

public void run() {

double avg = 0;

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

avg += arr[i];

avg = avg / n;

avgs[0] = avg;

}

});

Thread minThread = new Thread(new Runnable() {

@Override

public void run() {

int min = arr[0];

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

if (min > arr[i])

min = arr[i];

minmax[0] = min;

}

});

Thread maxThread = new Thread(new Runnable() {

@Override

public void run() {

int max = arr[0];

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

if (max < arr[i])

max = arr[i];

minmax[1] = max;

}

});

avgThread.start();

minThread.start();

maxThread.start();

try {

avgThread.join();

minThread.join();

maxThread.join();

}

catch (Exception e) {

}

System.out.println("Avarage Value :" + avgs[0] + "\n"+ "Minimum Value : " + minmax[0] + "\n" + "Maximum Value : " + minmax[1]);

}

}