**/\*\***

**\* @(#)HappyNumbers.java**

**\***

**\* HappyNumbers application**

**\***

**\* @Author : nmessa**

**\* @version 1.00 2019/10/11**

**\*/**

import java.util.\*;

public class HappyNumbers {

public static void main(String[] args) {

for (int i = 1; i <= 1000; i++)

{

if(isHappy(i))

{

System.out.println(i);

}

}

}

//This method returns true if number is happy

public static boolean isHappy(int n)

{

//Create an empty array list

ArrayList sad = new ArrayList();

while (n != 1)

{

//if the arrayList contains the number not a happy number

if (sad.contains(n))

{

return false;

}

//Add number to sad list and get a new value

else

{

sad.add(n);

n = calcValue(n);

//System.out.println(n);

}

}

//if n gets to 1

return true;

}

//This method calculates the value of a number by squaring each

//digit and adding them together

public static int calcValue(int number){

int temp, total = 0;

while (number > 0)

{

temp = number%10;

total += temp\*temp;

number /= 10;

}

return total;

}

}

**/\*\***

**\* @(#)PerfectNumber.java**

**\***

**\* PerfectNumber application**

**\***

**\* @Author: nmessa**

**\* @version 1.00 2019/10/10**

**\*/**

public class PerfectNumber {

public static void main(String[] args) {

for (int number = 1; number <= 10000; number++)

if (isPerfect(number))

System.out.print(number + " ");

}

//This method returns true if the number is perfect

//The sum of all divisors equals the number

public static boolean isPerfect(int n){

int total = 0;

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

{

if (n%i == 0)

total += i;

}

if (total == n)

return true;

else

return false;

}

}

/\*\*

\* @(#)MersennePrimes.java

\* MersennePrimes application

\* @Author: nmessa

\* @version 1.00 2019/10/10

\*/

import java.util.\*;

public class MersennePrimes {

public static void main(String[] args) {

int numbers[] = new int[8];

int mercennes[] = new int[8];

int count = 0;

for (int number = 2; number < 100;number++)

{

if (isPrime(number))

{

if (isMercenne(number))

{

numbers[count] = number;

int temp = 1;

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

temp \*= 2;

temp -= 1;

mercennes[count] = temp;

count++;

}

if(count == 8)

break;

}

}

for (int i = 0; i < numbers.length; i++)

{

System.out.println(numbers[i] + " " + mercennes[i]);

}

}

//This method returns true if the number is prime

public static boolean isPrime(int n){

for (int i = 2; i < (int)Math.sqrt(n)+1; i++) //more efficient loop

{

if (n%i == 0)

return false;

}

return true;

}

//This method returns true if the number is a Mersenne Prime

public static boolean isMercenne(int n)

{

int temp = 1;

//Calculate Mersenne number

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

temp \*= 2;

temp -= 1;

if (isPrime(temp))

{

return true;

}

else

{

return false;

}

}

}