Ejercicio 1

Dado un número natural n, indicar si es primo o no. El nombre de la función debe ser "esPrimo" y el tipo del valor de retorno deber ser Boolean.

Nota: Un número es primo cuando sólo es divisible por sí mismo y por la unidad.

PRE: n: número entero positivo mayor a 1 (1 no es primo por definición).

POST: Retorna un valor booleano:

True si el número es primo.

False si el número no es primo.

Ejercicio 2

Dado un número natural n, mostrar por pantalla el número primo más cercano que sea mayor apropio número n.

PRE: n: número entero positivo (incluido el cero).

POST: Retorna el primer número primo mayor a n.

Ejercicio 3

Usando sólo sumas y restas, calcular y mostrar por pantalla el resto y el cociente de la división entera de dos números naturales a y b.

PRE: a: número entero positivo (incluido el cero). b: número entero mayor a 0.

POST: Mostrar por pantalla los valores enteros del cociente y resto de la división entera.

Ejercicio 4

Dados dos números enteros a y b, obtener su máximo común divisor. El nombre de la función debe ser "calcularMCD" y el tipo del valor de retorno deber ser Integer.

PRE: a y b números enteros positivos (incluido el cero).

POST: Retorna un INTEGER que representa el MCD de los dos números.

Ejercicio 5

Diseñar una rutina que, dadas dos variables, numerador y denominador simplifique la fracción

numerador / denominador (sugerencia usar el algoritmo de Euclides).

PRE: numerador: número entero. Denominador número entero, excluido el cero.

POST: muestra por pantalla las formas mas simplificadas del numerador y denominador.

Ejercicio 6

Dado un número determinar si es capicúa. El nombre de la función debe ser “esCapicua” y el tipo del valor de retorno deber ser boolean.

Ejemplos de números capicúas: 24842, 11, 565, 1799971

PRE: n (o nombre del parámetro): número entero positivo.

POST: Retorna un valor booleano:

True si el número es capicúa.

False si el número no es capicúa.

Ejercicio 7

Listar por pantalla todos los números capicúas de 5 cifras (desde el 10001 en adelante).

PRE: no tiene.

POST: Listado por pantalla de todos los números capicúas entre el 10001 y el 99999.

Ejercicio 8

a) Escriba una función int maximoDelVector (int v[ ], int desde, int hasta) que devuelve el

máximo valor encontrado en el vector v considerando únicamente las posiciones entre desde y hasta.

PRE: desde >= 0, hasta >= 0, desde < largo del array , hasta < largo del array, desde <= hasta

POST: Retorna el máximo valor del array entre desde y hasta inclusive.

b) Escriba una función int posicionDelMaximoDelVector (int v[ ], int desde, int hasta) que

devuelve la posición del máximo valor encontrado dentro del vector v considerando únicamente las posiciones entre desde y hasta.

PRE: mismas que parte a.

POST: Retorna un entero que representa la posición el máximo valor del array entre desde y hasta inclusive.

Ejercicio 9

Escriba un procedimiento void ordenar (int v[ ], int n) que reciba un vector de n elementos y lo ordene.

PRE: n entero mayor a cero.

v debe contener al menos un valor entero.

POST: procedimiento ordena el vector dado (y lo muestra por pantalla).

Ejercicio 10

a) Escriba una función promedio que reciba un vector de N elementos y retorne el promedio.

b) Dado un vector de N elementos, hallar su mínimo, la cantidad de veces que se repite y la posición donde aparece por primera vez.

PRE: v debe contener al menos un valor numérico.

POST: calcula valor mínimo, cantidad de veces que se repite y primer índice de aparición. (los muestra por pantalla).

Ejercicio 11

Dados dos vectores ordenados v1 y v2 de dimensiones n1 y n2 respectivamente escribir una función que genere un tercer vector v3, también ordenado, que contenga los elementos de los dos anteriores.

PRE: v1 y v2 vectores que contengan al menos un valor, en caso de contener mas de uno, los mismos deben estar ordenados.

POST: retorna un vector v3 con los valores de v1 y v2 ordenados.

Ejercicio 12

Escribir un procedimiento que reciba un vector de N enteros y dos variables, max1 y max2,

cargue en ella los dos números mayores del vector (discutir posibles post--condiciones). necesito que me indiques las pre y post condiciones que deben cumplir esos algoritmos

PRE: v: vector de enteros con al menos dos elementos.

> max1 y max2 no pueden ser variables primitivas, en java no es posible cargarles valores ni mutarlas. Deberíamos aplicar alguna otra estrategia, como pasar algún objeto los cuales tengan como propiedades max1 y max2 o directamente pasar un array de enteros, y a la posición 0 cargarle el max1 y a la posición 1 el max2 por ejemplo. ¿qué otra cosa se les ocurre?