ISIS 1105 Diseño y Análisis de Algoritmos
Semestre 2022-20. Proyecto – PARTE 3
Entrega: Diciembre 7 de 2022 6:00PM (por Bloque Neon)

0 OBJETIVOS

- Diseñar soluciones computacionales para problemas.
- Estimar costos de las soluciones planteadas.
- Implementar soluciones.

Se premiarán las mejores soluciones y se castigarán las peores, en cuanto a eficiencia en tiempo y espacio.

1 CONDICIONES GENERALES

El proyecto se divide en tres partes independientes entre sí. Este documento describe la PARTE III. Cada parte contiene un problema a resolver mediante soluciones implementadas en `Java` o `Python`.

Para cada problema se pide:
- Descripción de la solución.
- Análisis temporal y espacial.
- Una implementación en Java o Python

2 DESCRIPCIÓN DEL PROBLEMA

A El salto divisor

Un robot que se encuentra ubicado en el punto 0 a lo largo de una línea recta infinita unidimensional. El robot puede hacer movimientos que incrementan su posición (no hay movimientos hacia atrás) según un entero positivo l_i. Este número representa la longitud del i-ésimo movimiento, el cual según el estudiante que programo el robot debe seguir las reglas del "salto divisor".

El "salto divisor" recibe un valor entero k>0 que limita los posibles valores de l_i así:

- La longitud del primer movimiento (l_0) deberá ser divisible por k.
- La longitud del segundo movimiento (l_1) deberá ser divisible por k+1.
- La longitud del tercer movimiento (l_2) deberá ser divisible por k+2.
- La longitud del cuarto movimiento (l_3) deberá ser divisible por k+3.
- ….. y así sucesivamente.

Por ejemplo, si k=2, entonces la secuencia de movimientos podría ser algo como: 0, 4, 7, 23, 88, ya que:

- 4-0=4 divisible por k
- 7-4=3 divisible por k+1
- 23-7=16 divisible por k+2
- 88-23=65 divisible por k+3

Problema

Dado dos enteros positivos m y k, su tarea es contar el número de formas para llegar al valor m, iniciando de 0,

Ejemplo:

Suponga m=5 y k=1,

- Maneras de llegar al punto 1: (0,1) Total:1
- Maneras de llegar al punto 2: (0,2) Total:1
- Maneras de llegar al punto 3: (0,1,3),(0,3) Total: 2
- Maneras de llegar al punto 4: (0,2,4),(0,4) Total: 2
- Maneras de llegar al punto 5: (0,1,5),(0,3,5) ,(0,5) Total: 3

La salida esperada por lo tanto seria: 3

3 ENTRADA Y SALIDA DE DATOS

En todas las soluciones que se presenten, la lectura de los datos de entrada se hace por la entrada estándar; así mismo, la escritura de los resultados se hace por la salida estándar.

Puede suponer que ninguna línea de entrada tiene espacios al principio o al final, y que los datos que se listan en cada línea están separados por exactamente un espacio.

A continuación, se establecen parámetros que definen su tamaño y formato de lectura de los datos, tanto de entrada como de salida.

Descripción de la entrada

La primera línea de entrada especifica el número de cados de prueba que contiene el archivo. El programa debe terminar su ejecución, una vez termine de resolver la cantidad de casos de prueba dados por este número.

Cada caso contiene una única línea con (2) enteros m, k ($1 \leq k \leq m \leq 10^5$).

Descripción de la salida

Se debe imprimir un número entero que representa el número de formas de llegar a m, iniciando en 0

Nota importante: Como se pueden obtener números muy grandes, para evitar desbordar la capacidad de números enteros, a cada suma que realice para calcular el valor objetivo se le debe aplicar la operación módulo 998244353.

Ejemplo de entrada / salida

Para el ejemplo realizado anteriormente, la descripción de la entrada/salida es la siguiente.

Entrada	Salida

3 5 1 4 2 9 7	3 1 0

Nota: Se van a diseñar casos de prueba para valores de m, y k muchos más grandes y dentro de los valores establecidos en el enunciado. Los casos mostrados en este documento son demostrativos de la estructura de entrada/salida esperada.

4 COMPRENSIÓN DE PROBLEMAS ALGORITMICOS

A continuación, se presentan un conjunto de escenarios hipotéticos que cambian el problema original. Para cada escenario debe contestar[1]: (i) que nuevos retos presupone este nuevo escenario -si aplica-?, y (ii) que cambios -si aplica- le tendría que realizar a su solución para que se adapte a este nuevo escenario?

ESCENARIO 1: Suponga que tiene un número limitado de movimientos t para llegar a un valor determinado m.

ESCENARIO 2: El robot no parte de cero si no de un número $p > 0$ primo.

Nota: Los escenarios son independientes entre sí.

5 ENTREGABLES

El proyecto puede desarrollarse por grupos de hasta tres estudiantes de la misma sección. La entrega se hace por bloque neon (una sola entrega por grupo de trabajo).

El grupo debe entregar, por bloque neon, un archivo de nombre `proyectoDalgoP3.zip`. Este archivo es una carpeta de nombre `proyectoDalgoP3`, comprimida en formato `.zip`, dentro de la cual hay archivos fuente de soluciones propuestas y archivos que documentan cada una de las soluciones.

5.1 Archivos fuente de soluciones propuestas

Todos los programas implementados en *Java* o en Python

Para el problema:

- Entregar un archivo de código fuente en *Java* (`.java`) o python (`.py`) con su código fuente de la solución que se presenta.

[1] NO tiene que implementar la solución a estos escenarios, el propósito es meramente analítico.

- Incluir como encabezado de cada archivo fuente un comentario que identifique el (los) autor(es) de la solución.
- Denominar `ProblemaP3.java` o `ProblemaP3.py` el archivo de la solución que se presente.

Nótese que, si bien puede utilizarse un *IDE* como *Eclipse* o Spyder durante el desarrollo del proyecto, la entrega requiere incluir solo un archivo por cada solución. El archivo debe poderse compilar y ejecutar independientemente (sin depender de ninguna estructura de directorios, librerías no estándar, etc.).

5.2 Archivos que documentan la solución propuesta

La solución al problema debe acompañarse de un archivo de máximo 3 páginas que la documente, con extensión `.pdf`. El nombre del archivo debe ser el mismo del código correspondiente (`ProblemaP3.pdf`).

Un archivo de documentación debe contener los siguientes elementos:

0 Identificación
Nombre de autor(es)
Identificación de autor(es)

1 Algoritmo de solución
Explicación del algoritmo elegido. Si hubo alternativas de implantación diferentes, explicar por qué se escogió la que se implementó.
Deseable:
Anotación (contexto, pre-, poscondición, …) para cada subrutina o método que se use.

2 Análisis de complejidades espacial y temporal
Cálculo de complejidades y explicación de estas. Debe realizarse un análisis para cada solución entregada.

3 Respuestas a los escenarios de comprensión de problemas algorítmicos.
Respuesta a las preguntas establecidas en cada escenario. NO tiene que implementar la solución a estos escenarios, el propósito es meramente analítico.

Téngase en cuenta que los análisis de 2 tienen sentido en la medida que la explicación de 1 sea clara y correcta. No se está exigiendo formalismo a ultranza, pero sí que, como aplicación de lo estudiado en el curso, se pueda describir un algoritmo de manera correcta y comprensible.

No describa un algoritmo con código GCL a menos que lo considere necesario para explicarlo con claridad. Y, si lo hace, asegúrese de incluir aserciones explicativas, fáciles de leer y de comprender.