UTEC
UNIVERSIDAD DE INGENIERÍA
Y TECNOLOGÍA

Algoritmos y Estructura de Datos

Examen Parcial
Pregrado
2021-0
Profesor Pedro Shiguihara
Lab 1.01

Indicaciones específicas:

- Esta evaluación contiene **??** páginas (incluyendo esta página) con **??** preguntas. El total de puntos son **??**.
- El tiempo límite para la evaluación es 110 minutos.
- Cada pregunta deberá ser respondida en un solo archivo con el número de la pregunta y tu código de estudiante. Por ejemplo:
  1. `p1.cpp`
  2. `p2.cpp`
  3. `p3.cpp`
  4. `p4.cpp`
- El envío de las soluciones es por www.gradescope.com
- Tome en cuenta que de no seguir de manera adecuada las indicaciones señaladas, la(s) pregunta(s) serán calificadas con 0.

Calificación:

Tabla de puntos (sólo para uso del professor)

Run LaTeX again to produce the table

1. (5 points) Objetivo: Implementar la solución de este problema con la estructura de datos cola basada en nodos enlazados. En caso sea necesario, se podrían usar estructuras auxiliares como listas. Un flujo de datos es una secuencia ordenada, continua y en tiempo real de elementos. Algunos ejemplos incluyen datos de sensores, tráfico de Internet, tickets financieros, subastas en línea y registros de transacciones, como registros de uso de la Web y registros de llamadas telefónicas. Del mismo modo, las consultas sobre transmisiones se ejecutan continuamente durante un período de tiempo y devuelven nuevos resultados de forma incremental a medida que llegan nuevos datos. Por ejemplo, un sistema de detección de temperatura de un almacén de fábrica puede ejecutar consultas como las siguientes.

   - Consulta-1: "Cada cinco minutos, recupere la temperatura máxima durante los últimos cinco minutos".
   - Consulta-2: "Devuelve la temperatura promedio medida en cada piso durante los últimos 10 minutos".

   Hemos desarrollado un sistema de gestión de flujo de datos llamado Argus, que procesa las consultas sobre los flujos de datos. Los usuarios pueden registrar consultas en Argus. Argus mantendrá las consultas ejecutándose sobre los datos cambiantes y devolverá los resultados al usuario correspondiente con la frecuencia deseada. Para Argus, usamos la siguiente instrucción para registrar una consulta:

   Registro Q-num Período (Ejm: Registro 2004 200)

   Qnum ($0 < Qnum \leq 3000$) es el número de ID de la consulta, y Periodo ($0 < Período <= 3000$) es el intervalo entre dos devoluciones consecutivas del resultado. Después de los segundos de período de registro, el resultado se devolverá por primera vez, y después de eso, el resultado se devolverá cada segundo de período. Aquí tenemos varias consultas diferentes registradas en Argus a la vez. Se confirma que todas las consultas tienen Qnum diferente. Su tarea es decirle a las primeras K consultas que devuelvan los resultados. Si dos o más consultas van a devolver los resultados al mismo tiempo, devolverán los resultados uno por uno en orden ascendente de Qnum.

   Sobre la entrada, la primera parte de la entrada son las instrucciones de registro para Argus, una instrucción por línea. Puede asumir que el número de instrucciones no superará las 1000 y que todas estas instrucciones se ejecutan al mismo tiempo. Esta parte termina con una línea '#'. La segunda parte es tu tarea. Esta parte contiene solo una línea, que es un entero positivo K (≤ 10000).

   Sobre la salida, se debe generar el Qnum de las primeras K consultas para devolver los resultados, un número por línea.

   Algunos ejemplos de diálogo de este programa serían:

   Listing 1: Ejemplo 1

```
Input:
Registro 2004 200
Registro 2005 300
#
```

```
5

Output:
2004
2005
2004
2004
2005
```

Listing 2: Ejemplo 2

```
Input:
Registro 1001 150
Registro 1002 200
Registro 1003 300
#
15

Output:
1001
1002
1001
1003
1002
1001
1001
1002
1003
1001
1002
1001
1003
1002
1001
```

Listing 3: Ejemplo 3

```
Input:
Registro 2100 20
Registro 2110 30
Registro 2120 40
#
20


Output:
```

```
2100
2110
2100
2120
2100
2110
2100
2120
2110
2100
2100
2110
2120
2100
2110
2100
2120
2100
2110
2100
```

Listing 4: Ejemplo 4

```
Input:
Registro 2050 40
Registro 2060 50
Registro 2070 60
#
10



Output:
2050
2060
2070
2050
2060
2050
2070
2060
2050
2070
```

2. (5 points) Objetivo: Implementar la solución de este problema con la estructura de

datos pila basada en nodos enlazados. En caso sea necesario, se podrían usar estructuras auxiliares como listas. Dom Cobb y su compañero Arthur realizan actividades ilegales al entrar en las mentes subconscientes de sus objetivos. Utilizan el sueño de dos niveles dentro de una estrategia de sueños para extraer información valiosa. Los soñadores se despiertan por un impacto repentino (una "patada") o pueden morir en el sueño. Ahora Dom quiere tu ayuda. Porque tiene una tarea compleja. Tiene que atravesar los sueños de muchas personas. Viaja del sueño de una persona al sueño de otra. Está demasiado consumido que ya no puede seguir si está despierto o en el sueño de otra persona. Te dará n consultas para procesar, cada una de las consultas estará en una de las siguientes formas:

"Sleep X": esto significa que la persona llamada X estará durmiendo y Dom entrará en el sueño de X, del sueño de la persona anterior (si lo hay).

"Kick": esto significa que la persona actual en cuyo sueño ha entrado Dom, se despertará y Dom volverá al sueño de la persona anterior de donde vino a este sueño. Si Dom ya no está en el sueño de nadie, ignora esta consulta.

"Test": esto significa que Dom quiere saber en qué sueño se encuentra en este momento. Tienes que escribir el nombre de la persona. Si Dom no está en el sueño de nadie en este momento, debe imprimir "No en un sueño" (sin las comillas).

Entrada: La primera línea contendrá un número entero n ($1 \leq n \leq 10000$), el número de consultas. Cada una de las siguientes n líneas contendrá una de las tres consultas mencionadas anteriormente. Para el caso de la consulta "Sleep X", X será una cadena compuesta solo por letras mayúsculas o minúsculas y no tendrá más de 15 caracteres.

Salida: Para cada una de las consultas de "Test", escriba el nombre de la persona en cuyo sueño Dom se encuentra en este momento exactamente como apareció en la entrada. Si Dom no está en el sueño de nadie, escriba la línea "No está en un sueño". Verifique la entrada y salida de muestra para obtener más detalles.

Algunos ejemplos de diálogo de este programa serían:

Listing 5: Ejemplo 1

```
Input:
20
Sleep Dom
Sleep Sakin
Test
Sleep Asif
Sleep Mushfiq
Test
Kick
Test
Sleep Shafi
Test
Kick
Test
```

```
Kick
Test
Kick
Test
Kick
Test
Kick
Test


Output:
Sakin
Mushfiq
Asif
Shafi
Asif
Sakin
Dom
No esta en un sue o
No esta en un sue o
```

Listing 6: Ejemplo 2

```
Input:
25
Sleep Dom
Sleep Carlos
Test
Sleep Maria
Test
Sleep Ana
Test
Kick
Test
Sleep Andres
Test
Sleep Cindy
Test
Kick
Test
Kick
Test
Kick
Test
Kick
```

```
Test
Kick
Test
Kick
Test


Output:
Carlos
Maria
Ana
Maria
Andres
Cindy
Andres
Maria
Carlos
Dom
No esta en un sue o
No esta en un sue o
```

Listing 7: Ejemplo 3

```
Input:
10
Test
Kick
Test
Sleep Dom
Test
Sleep Marco
Test
Test
Kick
Test



Output:
No esta en un sue o
No esta en un sue o
Dom
Marco
Marco
Dom
```

Listing 8: Ejemplo 4

```
Input:
12
Kick
Test
Sleep Dom
Sleep Carl
Test
Sleep Ana
Test
Kick
Test
Kick
Test
Kick




Output:
No esta en un sue o
Carl
Ana
Carl
Dom
```

3. (5 points) Objetivo: Implementar la solución de este problema con la estructura de datos set con open addressing. En caso sea necesario, se podrían usar estructuras auxiliares como listas. Algunos amigos tienen el mismo hobby, coleccionar sellos. Érase una vez que decidieron hacer una exposición. La exposición les trajo algo de dinero y ahora no saben cómo repartir sus ingresos. Decidieron dividir su dinero de acuerdo con esta regla: "El porcentaje del ingreso total que obtendrá su i-ésimo amigo es igual a la parte del tipo de su sello único". El tipo de sello se llama único si y solo si este tipo de sellos es propiedad de una sola persona.

   Entrada La primera línea contiene el entero K ($0 < K \leq 100$), es el número de pruebas. Cada caso de prueba se describe con un entero positivo N ($0 < N \leq 50$), es el número de amigos. Luego sigue N líneas con números enteros. Cada línea corresponde a una colección de sellos de amigos. El primer número entero de la línea es M, el número de sellos que posee una persona ($0 < M \leq 50$). Luego viene M enteros A ($0 \leq A \leq 10000$) — tipos de estampillas.

   Salida Para cada caso de prueba, formatear de esta forma: "Caso i: a1% a2% a3% ... an%". Donde i es un número de prueba y un porcentaje de los ingresos que se destina al i-ésimo amigo. Los porcentajes de salida deben redondearse a dos decimales.

Algunos ejemplos de diálogo de este programa serían:

Listing 9: Ejemplo 1

```
Input:
1
3
3 1 2 3
2 4 5
3 4 2 6


Output:
Caso 1: 50.00% 25.00% 25.00%
```

Listing 10: Ejemplo 2

```
Input:
1
4
3 2 5 6
4 1 5 7 8
3 2 4 6
6 3 5 6 7 8 9

Output:
Caso 1: 0.00% 25.00% 25.00% 50.00%
```

Listing 11: Ejemplo 3

```
Input:
1
3
4 2 5 6 8
3 1 2 3
5 3 4 6 7 9

Output:
Caso 1: 33.33% 16.67% 50.00%
```

Listing 12: Ejemplo 4

```
Input:
1
4
5 2 4 6 7 9
3 5 8 9
4 1 7 6 2
```

```
2 3 7

Output:
Caso 1: 20.00% 40.00% 20.00% 20.00%
```

4. (5 points) Objetivo: Implementar la solución de este problema con la estructura de datos diccionario con separate chaining.

   Los estudiantes "Frosh" que comienzan sus estudios en Waterloo tienen intereses diversos, como lo demuestra su deseo de tomar varias combinaciones de cursos entre los disponibles. Los encargados de matrícula se sienten incómodos con esta situación y, por lo tanto, desean ofrecer un premio de conformidad a los Frosh que elijan la combinación o las combinaciones de cursos más populares. ¿Cuántos Frosh ganarán el premio?

   Entrada: La entrada consta de varios casos de prueba seguidos de una línea que contiene "0". Cada caso de prueba comienza con un número entero $1 \leq n \leq 10000$, el número de alumnos Frosh. Para cada frosh, sigue una línea que contiene los códigos de curso para cinco cursos distintos donde se matriculará el frosh. Cada código de curso es un número entero en el rango entre 100 y 499. La popularidad de una combinación es el número de frosh que seleccionan exactamente la misma combinación de cursos. Una combinación de 5 cursos se considera más popular si ninguna otra combinación de 5 cursos tiene mayor popularidad. Puede haber más de una combinación de 5 cursos populares.

   Salida: Para cada línea de entrada con "n" frosh, se debe generar una sola línea que indique el número total de frosh que han tomado la combinación de cursos más popular.

   Algunos ejemplos de diálogo de este programa serían:

Listing 13: Ejemplo 1

```
Input:
3
100 101 102 103 488
100 200 300 101 102
103 102 101 488 100
3
200 202 204 206 208
123 234 345 456 321
100 200 300 400 444
0

Output:
2
3
```

Listing 14: Ejemplo 2

```
Input:
4
```

```
200 201 203 204 360
200 204 206 208 210
206 210 300 320 360
203 204 360 201 210
0

Output:
4
```

Listing 15: Ejemplo 3

```
Input:
3
300 302 304 305 308
300 305 308 400 410
305 308 302 304 300
0

Output:
2
```

Listing 16: Ejemplo 4

```
Input:
5
120 122 126 140 160
122 140 150 180 200
120 140 160 180 200
150 180 200 140 122
126 160 122 140 120
0

Output:
4
```