Reto II: Problema de las cifras

Francisco David Charte Luque Ignacio Cordón Castillo
Mario Román García

TDA Cuenta

Cuenta
+ primero + segundo + operador + resultado

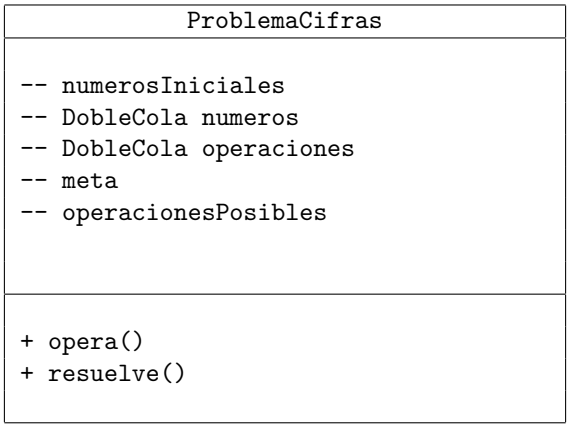
- `primero` Número entero, representando el primer operando.
- `segundo` Número entero, que representa el segundo operando.
- `operador` Carácter que corresponde a la operación realizada sobre los números.
- `resultado` Número entero, resultado de realizar la operación sobre `primero` y `segundo`.

TDA DobleCola

DobleCola
-- double_queue
+ push_back() + pop_back() + push_front() + pop_front() + front() + back()

- `double_queue` Vector de objetos `T` que es usado por el TDA `DobleCola` como una doble cola enlazada (en la que se puede suprimir e introducir elementos por ambos lados)
- `push_back()` Introduce un elemento `T` en la cola de `double_queue`
- `pop_back()` Saca un elemento `T` de la cola de `double_queue`
- `push_back()` Introduce un elemento `T` en el frente de `push_back()`
- `pop_back()` Saca un elemento `T` del frente de `push_back()`

TDA ProblemaCifras



- `numerosIniciales` Conjunto que almacena los enteros a partir de los que se pretende obtener meta
- `numeros` Instancia de `DobleCola` en la que se almacenan los números con los que se opera en cada llamada a `resuelve()`
- `operaciones` Instancia de `DobleCola` en la que se almacenarán las operaciones realizadas hasta llegar a meta, o a una aproximación a meta
- `meta` Entero positivo de 3 cifras a aproximar, y en caso de ser posible, hallar de forma exacta mediante operaciones sobre
- `operacionesPosibles` Conjunto que contiene todas las operaciones posibles aplicables

Algoritmo 1 ALGORITMO DE CÁLCULO DEL NÚMERO DE 3 CIFRAS

Entrada:

`meta`, número a aproximar
`numeros`, enteros aleatorios iniciales del conjunto
`size`, número de posiciones de la lista `numeros`

Salida: **true** si logramos alcanzar exactamente `meta`

1: **si** `size` es menor a 2 **entonces**
2: **devolver** **false**
3: **fin si**
4: **para** `i` desde la primera posición de la lista, hasta la anterior a la última (`size`)
5: Tomamos `a` como el número ocupando la posición i-ésima de la lista
6: **si** $a \neq 0$ **entonces**
7: sobreescribimos la posición que ocupaba `a` con lo que haya en la última posición de la lista
8: **en otro caso**
9: Avanzamos a la iteración siguiente
10: **fin si**
11: **para** `j` desde `i` hasta `size-1`
12: Tomamos `b` como el número que ocupa la posición j-ésima de la lista
13: **si** $b \neq 0$ **entonces**
14: sobreescribimos la posición que ocupaba `b` con lo que haya en la penúltima posición de la lista
15: **en otro caso**
16: Avanzamos a la iteración siguiente
17: **fin si**
18: **para** operación `OP` en $/, -, +, *$
19: Tomamos `c` como el mayor de `a` y `b`, y `d` como el menor de ambos
20: **fin para**
21: **fin para**
22: **fin para**