Insereix llista W20168_ca

Volem afegir el mètode **`insereix_llista_posicio(lst,pos)`** a la classe `Llista`.
Suposem que tenim una llista no buida `ll` (instància de la classe `Llista`), una altra llista `lst` (també instància de la classe `Llista`) i una posició `pos` *correcte* de la llista `ll` (és a dir $0 \leq$ `pos` $<$ `ll.mida()`).
El resultat d'invocar el mètode `ll.insereix_llista_posicio(lst,pos)` és:

- `ll` creix ja que s'insereix la llista `lst` tot just *abans* de la posició `pos` de la llista `ll`.
- La llista `lst` queda buida. És a dir, després de la crida al mètode, `lst.buida()` retorna `True`.

Per exemple, si la llista `ll` és 2 90 3 8 7 5 4 2 87 67, la llista `lst` és −100 23 41 i `pos` és 3, la invocació `ll.insereix_llista_posicio(lst,pos)` deixarà la llista `ll` com 2 90 3 −100 23 41 8 7 5 4 2 87 67 i la llista `lst` quedarà buida (`lst.buida()` retorna `True`)

Precondició

La llista sobre la que s'invoca el mètode (el *paràmetre implícit*) no pot ser buida.
La llista paràmetre `lst` és instància de la classe `Llista`
La posició `pos` és *correcte* respecte a la llista sobre la que s'invoca el mètode. És a dir, si fem `ll.insereix_llista_posicio(lst,pos)`, aleshores $0 \leq$ `pos` $<$ `ll.mida()`.

Entrada

Primer hi ha un natural n, que ens indica el nombre d'elements de la llista. Després apareixen n nombres enters, que formen la llista inicial. Es garanteix que $n > 0$.
Després trobem un natural m, que ens indica el nombre d'elements de la llista a inserir. Tot seguit hi ha m nombres enters, que formen la llista que volem inserir a la llista anterior.
Finalment trobem un natural p, que és la posició on volem inserir la llista. És garanteix que $0 \leq p < n$.
Vegeu els exemples que formen el joc de proves públic.

Sortida

La representació textual de la llista inicial després de fer la corresponent inserció de la llista indicada a l'entrada a la posició corresponent. Recordem que la representació textual de les llistes s'obté fent `print(`*lst*`)` (on *lst* és una instància de la classe `Llista`).

Observacions

Dins el codi del mètode demanat **NO** podeu fer servir cap mètode de la classe `Llista` (hi ha una excepció: no és imprescindible, però podeu fer servir el mètode `__init__` per buidar la llista paràmetre). Heu de manipular exclusivament les instàncies de `_Node` que formen la implementació de les llistes.
Vegeu els comentaris del mètode per completar al fitxer **`code.py`**.

Heu de baixar-vos el fitxer **`code.py`** (icona de la serp). Aquest fitxer és un programa amb **tot** el que cal per executar els jocs de prova públics. Només falta, clar, el mètode que us demana l'enunciat. Aquest fitxer l'heu de completar amb el codi que falta, i això, **tot**, és el que heu d'enviar al Jutge com a solució.

L'eficiència i la qualitat de la solució es tindran en compte a la correcció manual.

Exemple d'entrada 1

```
7
-50 -37 81 -60 -61 3 98
6
-51 -88 -87 30 -53 38
4
```

Exemple de sortida 1

```
-50 -- -37 -- 81 -- -60 -- -51 -- -88 -- -87 -- 30 -- -
```

Exemple d'entrada 2

```
2
98 77
5
-86 -32 -81 -92 -39
1
```

Exemple de sortida 2

```
98 -- -86 -- -32 -- -81 -- -92 -- -39 -- 77
```

Exemple d'entrada 3

```
5
-66 -35 -24 80 45
7
-40 71 19 -21 54 46 6
4
```

Exemple de sortida 3

```
-66 -- -35 -- -24 -- 80 -- -40 -- 71 -- 19 -- -21 -- 54
```

Exemple d'entrada 4

```
8
95 -62 -27 51 -87 76 44 58
7
94 -89 46 -82 41 38 62
7
```

Exemple de sortida 4

```
95 -- -62 -- -27 -- 51 -- -87 -- 76 -- 44 -- 94 -- -89
```

Exemple d'entrada 5

```
4
6 80 3 58
6
-59 -30 -15 37 92 10
2
```

Exemple de sortida 5

```
6 -- 80 -- -59 -- -30 -- -15 -- 37 -- 92 -- 10 -- 3 --
```

Exemple d'entrada 6

```
1
-100
2
3 4
0
```

Exemple de sortida 6

```
3 -- 4 -- -100
```

Informació del problema

Autoria: Jordi Delgado

Generació: 2026-01-25T13:15:36.065Z