Konputaziorako Sarrera
Ohiko deialdia – 2024/2025

Arauak

- Ezin da inolako materialik (libururik, apunterik, etab.) ezta gailu elektronikorik (ordenagailua, mugikorra, kalkulagailua, etab.) erabili.

1. (3 puntu) **1. Ariketa**

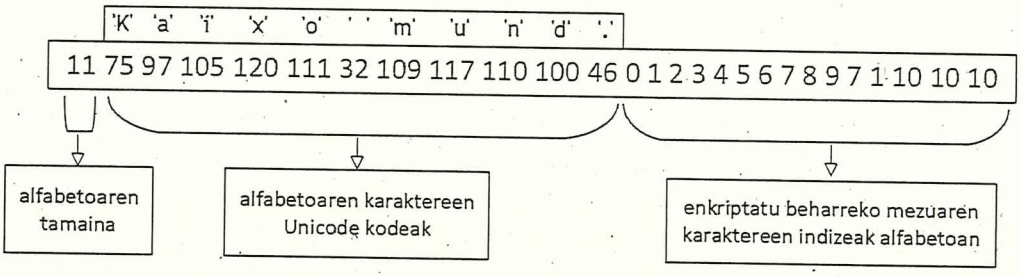
Python-eko `ord()` eta `chr()` funtzioek karaktereen Unicode adierazpenarekin lan egiteko balio dute. `ord()`-ek karaktere bat hartzen du argumentu moduan eta zenbaki oso bat buelatzen du, karakterearen Unicode kodea. Bestalde, `chr()`-ek zenbaki oso bat hartzen du argumentu moduan eta kode horri dagokion karakterea bueltazen du. Adibidez:

```
ord("K") → 75
ord("a") → 97
ord(".") → 46
```

```
chr(75) → "K"
chr(97) → "a"
chr(46) → "."
```

Kode hauetatik abiatuz, mezu bat (karaktere kate bat) enkriptatzeko hurrengo prozesua diseinatu da. Mezu enkriptatuta hutsuneen bidez banatutako zenbakiez osotuta egongo da. Lehenengo zenbakia jatorrizko mezuaren alfabetoaren tamaina izango da, hau da, karaktere ezberdinen kopurua. Ondoren, alfabetoaren karaktereen Unicode kodeak agertzen dira, `ord()` funtzioaren bidez lor daitezkeenak. Amaitzeko, mezuaren karaktere bakoitzaren indizea alfabetoan agertzen da. Adibidez,

```
mezua = "Kaixo mundua..."  ->  alfabetoa = "Kaixo mund."
```



(a) `alfabetoa_sortu(kk)` funtzioa sortu. Honek sarrerako katean agertzen diren karaktereak dauzkan karaktere kate bat bueltatuko du, baina karaktere bakoitza aldi bakar batean agertuko da.

(b) `enkriptatu(kk)` funtzioa diseinatu. Honek sarrerako karaktere katea aurreko enkriptazio prozesua jarraituz enkriptatuko du, eta karaktere kate enkriptatua bueltatu. Adibidez:

```
enkriptatu("Kaixo mundua") ->
"11 75 97 105 120 111 32 109 117 110 100 46 0 1 2 3 4 5 6 7 8 9 7 1 10 10 10"
```

(c) `desenkriptatu(kk)` funtzioa diseinatu. Honek aurreko funtzioak bueltatutako kate enkriptatu bat jaso eta jatorrizko mezua bueltatuko du. Hau egiteko, lehenengo pausua kate enkriptatuatik abiatuz alfabetoa sortzea da, `chr()` erabiliz. Ondoren, jatorrizko mezuaren karaktereak indizeen bidez lortu daitezke. Adibidez:

```
desenkriptatu("11 75 97 105 120 111 32 109 117 110 100 46 0 1 2 3 4 5 6 7 8 9 7 1
    10 10 10") -> "Kaixo mundua"
```

2. (3 puntu) **2. Ariketa**

Izan bedi zenbaki osoz osatutako `z` zerrenda. Hiru funtzio diseinatu:

(a) `bilatu(z, azpi_z)` funtzioa idatzi. Honek, `z`-z gain, beste zerrenda bat hartuko du argumentu moduan, `azpi_z`. Funtzioak `azpi_z` azpisegidaren lehenengo agerpenaren posizioa `z`-n bueltatuko du. Azpisegida ez aurkituz gero, `-1` izango da emaitza. Beraz, funtzio hau karaktere kateen `find()` metodoaren antzekoa da, baina zerrendetarako. Adibidez:

```
bilatu([1, 2, 3, 4], [2, 3]) -> 1
bilatu([1, 2, 3, 2, 3, 4], [2, 3, 4]) -> 3
bilatu([1, 2, 1, 2, 1, 2], [1, 2]) -> 0
bilatu([1, 2, 3, 4], [1, 4]) -> -1
```

(b) Aurreko funtzioa erabiliz, `lehenengo_agerpena_ezabatu(z, azpi_z)` funtzioa diseinatu. Honek `azpi_z` azpisegidaren lehenengo agerpena ezabatuko du `z`-tik. `azpi_z` `z`-ren barruan ez egonez gero, `z` ez da aldatu behar. Edozein kasutan, funtzioak ez du baliorik bueltatuko.

(c) Aurreko funtzioa(k) erabiliz, `agerpen_guztiak_ezabatu(z, azpi_z)` funtzioa diseinatu, `azpi_z` azpisegidaren agerpen guztiak `z`-tik ezabatuko dituena. Funtzioak ez du baliorik bueltatuko.

3. (4 puntu) **3. Ariketa**

Demagun matrize bat testu fitxategi batean gordetzen dela, hurrengo formatuarekin: lehenengo lerroan bi zenbaki agertuko dira, matrizearen lerro kopurua eta zutabe kopurua, hutsune batez bananduta. Hurrengo lerroetan matrizearen datuak agertzen dira, betiere hutsuneen bidez bananduta. Adibidez:

```
3 4
1 0 0 0
0 1 2 0
0 0 0 1
```

(a) `matrizea_jaso(bideizen)` funtzioa diseinatu, `bideizen` fitxategian aurreko formatuan gordeta dagoen matrizea irakurri eta **hiztegi** bat bueltatzen duena, matrizea adierazteko. Matrizearen giltzak matrizearen posizioak adierazten duten tuplak izango dira, eta balioak posizio bakoitzari dagokion balioa. Ez da balio nulurik gordeko. Aurreko adibidearekin jarraituz, hiztegi hau bueltatu beharko luke funtzioak:

```
M={(0,0): 1, (1,1): 1, (1,2): 2, (2,3): 1}
```

(b) `matrizeak_batu(M1, M2)` funtzioa idatzi, bi matrize adierazten duten hiztegiak jaso eta hauen batura bueltatzen duena, beste hiztegi bat erabiliz. Ez da balio nulurik egongo, ez sarrerako hiztegietan, ez irteerakoan.

(c) `hiztegitik_zerrendetara(M)` funtzioa idatzi. Honek aurreko atalen moduko hiztegi bat jasoko du argumentu moduan, matrize bat adierazten duena. Irteera matrize hori bera, baina zerrenden zerrenda baten moduan adierazia izango da. Hau da, a) ataleko hiztegia erabiliko bagenu, bueltatu beharrekoa hurrengoa litzateke:

```
hiztegitik_zerrendetara(M) -> [[1, 0, 0, 0], [0, 1, 2, 0], [0, 0, 0, 1]]
```

(d) Argumentu moduan hiztegi baten bidez adierazitako `M` matrize bat eta irteerako bi fitxategien bideizenak jasotzen dituen funtzioa sortu. Honek bi fitxategi sortu beharko ditu: lehenengoa `M` matrizearekin, eta bigarrena bere irauliarekin. Ariketaren hasieran deskribatutako formatua erabili. Adibidez:

`M`-ri dagokion irteera fitxategia:

```
3 4
1 0 0 0
0 1 2 0
0 0 0 1
```

`M`-ren irauliari dagokion irteera fitxategia:

```
4 3
1 0 0
0 1 0
0 2 0
0 0 1
```