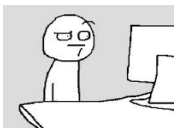


Oinarrizko Programazioa.
4. laborategia. Sekuentziak (Adaz eta Pythonez)

Izena:____Lucia Del Rio, Ainhoa Tomas_____Data:__2023/10/3_____

Laborategi honetan sekuentziak landuko dira.

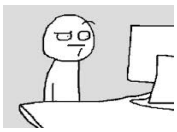
1. ariketa

10 zenbaki osoz osatutako sekuentzia bat edukita, sekuentzian dagoen azken zenbaki bikoitia eta berau aurkitzen den posizioa inprimatzeko programa idatz ezazue.

1. Espezifikazioa
Sarrera: *10 zenbaki oso sekuentzia*
Aurre: *sekuentzia 10 zenbakiz osatuta dago, ez bat gutxiago, ez bat gehiago.*
Irteera: *zenbaki osoko bi*
Post: *balioa1 zerrendako azken zenbaki bikoitia izango da eta balioa2 bera aurkitzen den posizioa. Balioa2 zenbakiak 0 balioa izango du baldin eta zerrendan ez badago zenbaki bikoitirik.*

2. Proba kasuak

sekuentzia	**emaitza**
3, 7, 4, 5, 9, 3, 1, 8, 5, 11	8, 8
1, 3, 5, 7, 9, 1, 5, 1, 3, 5	0, 0
3, 7, 9, 4, 5, 1, 7, 9, 3, 11	4, 4

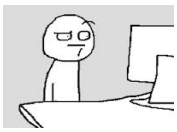


3. Algoritmoa

```
Sekuentzia1: 10 integer;
Bik, n: integer;
Bik ←0;
n←0;
Hasiera
        Irakurri_sekuentzia(sekuentzia1);
        Hasieran_jarri(sekuentzia1);
        Errepikatu(sekuentziatik_kanpo(sekuentzia1)) bete arte
                Baldin (egungo_elementua(sekuentzia1) rem 2 = 0) orduan
                        Bik ←egungo_elementua(sekuentzia1);
                        Gorde(sekuentzia1, n);
                Amaitu_baldin;
                Aurrera_egin;
        Amaitu_errepikatu;
        Idatzi (bik, “,” n);
Amaiera
```

4. Simulazioa

BUELTA KOP	**SARTU?**	**BIK**	**n**	**Irteera**
HASIERAKETAK	-	0	0	0,0
1	BAI	0	0	0,0
2	BAI	0	0	0,0
3	BAI	0	0	0,0
4	BAI	4	4	4,4
5	BAI	4	4	4,4
6	BAI	4	4	4,4
7	BAI	4	4	4,4
8	BAI	4	4	4,4
9	BAI	4	4	4,4
10	BAI	4	4	4,4
11	EZ			



2. ariketa

10 zenbaki osoz osatutako sekuentzia bat edukita, sekuentzian dauden elementuen artean, sekuentziako azken elementuagatik zatigarriak direnak inprimatzeko algoritmoa idatz ezazu.

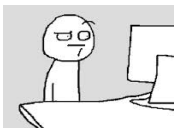
1. Espezifikazioa
Sarrera: *10 zenbaki oso sekuentzia*
Aurre: *sekuentzia 0 baino handiagoak diren 10 zenbakiz osatuta dago, ez bat gutxiago, ez bat gehiago.*
Irteera: *Zenbaki osoko bat edo gehiago.*
Post: *irteerako zenbaki bakoitza / sekuentziako azken balioa, hondarra = 0 da.*

2. Proba kasuak

sekuentzia	**emaitza**
3, 7, 4, 5, 9, 3, 1, 8, 5, 2	4, 8, 2
1, 3, 5, 7, 9, 1, 5, 1, 3, 11	11
3, 7, 9, 4, 5, 1, 7, 9, 3, 1	3, 7, 9, 4, 5, 1, 7, 9, 3, 1

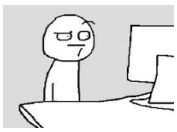
3. Algoritmoa

```
Sekuentzia1: 10 integer;
Zatitzailea: integer;
Hasiera
        Irakurri_sekuentiza(sekuentzia1);
        Amaieran_jarri(sekuentzia1);
        Zatitzailea←egungo_elementua(sekuentiza1);
        Hasieran_jarri(sekuentzia1);
        Errepikatu (sekuentziatik_kanpo(sekuentzia1))bete arte
                Baldin (egungo_elementua(sekuentiza1) rem zatitzailea = 0) orduan
                        Idatzi (egungo_elementua(sekuentzia1));
                Aurrera_egin(sekuentzia1);
        Amaitu_errepikatu;
Amaiera;
```



4. Simulazioa

BUELTA KOP	**SARTU?**	**zatitzailea**	**Egungo_zenbakia**	**Irteera**
HASIERAKETAK	-	2	2	-
1	BAI	2	3	-
2	BAI	2	7	-
3	BAI	2	4	4
4	BAI	2	5	4
5	BAI	2	9	4
6	BAI	2	3	4
7	BAI	2	1	4
8	BAI	2	8	4,8
9	BAI	2	5	4,8
10	BAI	2	2	4,8,2
11	EZ			



3. ariket8

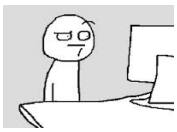
10 zenbaki osoz osatutako sekuentzia bat edukita, sekuentziako elementu guztiak ordenatuta dauden ala ez esango digun algoritmoa idatz ezazue.

1. Espezifikazioa
Sarrera: *10 zenbaki oso sekuentzia*
Aurre: *sekuentzia 10 zenbakiz osatuta dago, ez bat gutxiago, ez bat gehiago.*
Irteera: *Mezu bat*
Post: *Pantailan "sekuentzia ordenatuta dago" idatziko da baldin eta elementu* ***guztiak*** *ordenatuta badaude, bestela "sekuentzia EZ dago ordenatuta" idatziko da*

2. Proba kasuak
1, 2, 5, 5, 8, 9, 10, 12, 25, 67 → Bai, ordena jarraitzen du
4 ,2 ,5, 1, 8, 9, 10 ,12, 25, 67 → Ez, ez dago ordenatuta
1, 2, 5, 7, 8, 9, 10, 12, 25, 3 → Ez, ez dago ordenatuta
20, 14, 12, 8, 7, 6, 4, 3, 2, 1 → Bai, ordena jarraitzen du

3. Algoritmoa

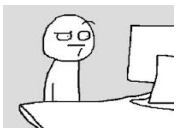
```
Sekuentzia1: 10 integer;
Aux: integer;
Hasiera
        Irakurri_sekuentiza(sekuentzia1);
        Hasieran_jarri(sekuentzia1);
        Aux←egungo_elementua(sekuentzia1);
        Aurrera_egin;
        Baldin (aux <egungo_elementua(sekuentiza1)) orduan
                Hasieran_jarri(sekuentzia1);
                Errepikatu (sekuentziatik_kanpo(sekuentzia1))bete arte
                        Aux←egungo_elementua(sekuentzia1);
                        Aurrera_egin;
                        Baldin(aux <= egungo_elementua(sekuentzia1)) orduan
                                Idatzi("Bai, ordenatuta dago");
                        Bestela
                                Idatzi("ez dago ordenatuta")
                        Amaitu_baldin;
                Amitu_errepikatu;
        bestela
                amaieran_jarri(sekuentzia1);
                Errepikatu (sekuentziatik_kanpo(sekuentzia1))bete arte
                        Aux←egungo_elementua(sekuentzia1);
                        Atzera_egin;
                        Baldin(aux <= egungo_elementua(sekuentzia1)) orduan
                                Idatzi("ordenatuta dago");
```



```
                    Bestela
                        Idatzi("ez dago ordenatuta")
                    Amaitu_baldin;
                Amitu_errepikatu;
            Amaitu_baldin;
Amaiera;
```

4. Simulazioa

BUELTA KOP	**SARTU 1.errepikapena?**	**SARTU 2.errepikapena?**	**aux**	**Egungo_elemen.**	**Irteera**
HASIERAKETAK	-	-	1	1	-
1	BAI	EZ	1	2	Bai, ordenatuta dago.
2	BAI	EZ	2	5	Bai, ordenatuta dago
3	BAI	EZ	5	5	Bai, ordenatuta dago
4	BAI	EZ	5	8	Bai, ordenatuta dago
5	BAI	EZ	8	9	Bai, ordenatuta dago
6	BAI	EZ	9	10	Bai, ordenatuta dago
7	BAI	EZ	10	12	Bai, ordenatuta dago
8	BAI	EZ	12	25	Bai, ordenatuta dago
9	BAI	EZ	25	67	Bai, ordenatuta dago
10	EZ	EZ			Bai, ordenatuta dago



4. ariketa

Eskatu erabiltzaileari zenbaki oso bat (balio >0) eta irudika ezazu hurrengo grafikoa algoritmo bidez:

N=5 balitz

```
*0000
**000
***00
****0
*****
```

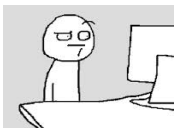
1. Espezifikazioa
Sarrera: zenbaki oso bat
Aurre: 0 baino handiagoa den zenbaki oso bat
Irteera: matrize bat
Post: Pantailan 0z eta *z osatutako matrize bat agertuko da, non baldin eta errenkada zutabea baino txikiago edo berdina izanez gero, *a agertuko da, bestela 0 bat.

2. Proba kasuak
N=3 balitz

```
*00
**0
***
```

N=5 balitz

```
*0000
**000
***00
****0
*****
```

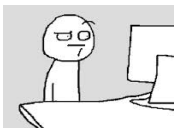


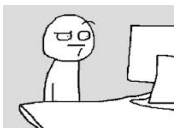
3. Algoritmoa

```
Zenbakia, errenkada, lerroa: ineteger;
Idatzi("sartu zenbaki bat");
Irakurri(zenbakia);
Lerroa←1;
Errenkada←1;
Errepikatu (lerroa > zenbakia) bete arte
        Bitartean (errenkada <= zenbakia)
                Baldin(errenkada <= lerroa) orduan
                        Idatzi("*")
                Bestela
                        Idatzi("0")
                Amaitu_baldin;
                Errenkada←errenkada +1;
        Amaitu_bitartean;
        Idatzi("\n");
        Lerroa←lerroa+1;
Amaitu_errepikatu;
```

4. Simulazioa

zenbakia	***errenkada***	***lerroa***	***pantaila***
3	1	1	*
3	2	1	*0
3	3	1	*00
3	1	2	*00 *
3	2	2	*00 **
3	3	2	*00 **0
3	1	3	*00 **0 *
3	2	3	*00 **0 **
3	3	3	*00 **0 ***





5. ariketa

Orain gure lehenengo programa egikarituko dugu programazio-lengoaia batean. Laborategietan beharko ditugun konpiladore eta interprete guztiak egongo dira (alegia, ADA konpiladorea eta Python interpretea). Programatzeko erabiliko dugun testuinguru grafikoa ere bertan egongo da (GPS). Ala ere Pythonez programatzeko CodeSkultor erabiltzea da errazena, ez baita instalatu behar. Erabiltzeko joan hurrengo estekara:

https://py3.codeskulptor.org

kaixoMundua programa egikarituko dugu bai ADA-z eta Python-ez. (“Kaixo mundua” mezua pantailan inprimatzen duen programa)

6. ariketa

Erabiltzaileari >0 den zenbaki oso bat eskatuko dion programa bat idatziko dugu. Programa horren helburua, erabiltzaileak sartutako zenbakiak zenbat digitu bakoiti dituen kalkulatzea izango da.

1. Espezifikazioa

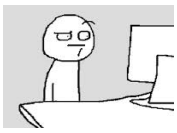
Sarrera : *zenbaki bat*
Aurre: *zenbaki oso bat >0*
Irteera: *zenbaki oso bat*
Post: *pantailatik >=1 den balio bat inprimatuko da. Balio hori, sarrerako zenbakiaren digitu bakoitien kopurua izango da.*

2. Proba kasuak

12345 → zuk sartutako zenbakiak 3 digitu bakoiti ditu.
22446 → zuk sartutako zenbakiak 0 digitu bakoiti ditu.

3. Algoritmoa
[Nahi izanez gero pseudokodea jarri dezakezue hemen, baina ez da derrigorrezkoa. Derrigorrez entregatu beharrekoa kodea (bai Adaz baita pyhonez) duten fitxategiak, digitu_bakoitiak_kontatu.adb eta digitu_bakoitiak_kontatu.py, izango dira]

4. Simulazioa
12345 → zuk sartutako zenbakiak 3 digitu bakoiti ditu.
22446 → zuk sartutako zenbakiak 0 digitu bakoiti ditu.



7. ariketa

Erabiltzaileari batez hasten den zenbaki bitar bat eskatu (zenbaki oso gisa gordeko dena). Kalkulatu zenbaki horren baliokide hamartarra.

1. Espezifikazioa

Sarrera : *zenbaki bat*
Aurre: *1z hasten den zenbaki bitar bat (hau da 0z eta 1z soilik osatua)*
Irteera: *zenbaki oso bat*
Post: *Sarrerako zenbaki bitarraren balio hamartarra. Gogoratu zenbaki bitarretan zifra bakoitzak bere pisua duela (zifra* $2^{posizioa}$). Adibidea*

$10010= 1*2^4+0*2^3+0*2^2+1*2^1+0*2^0= 18$
*ADAz berreketa: oinarria**exp*
*Python-ez berreketa: oinarria**exp*

2. Proba kasuak

101 → Zenbakia hamartarrez honakoa da: 5
1 → Zenbakia hamartarrez honakoa da: 1

3. Algoritmoa

4. Simulazioa
101 → Zenbakia hamartarrez honakoa da: 5
1 → Zenbakia hamartarrez honakoa da: 1