Oinarrizko Programazioa - 7. laborategia
Azterketetako ariketak

OHARRA:

1. Proposatzen den ariketa bateko txantiloiren bat edo probetarako programa faltaz gero, zuek sortu eta eraiki beharko duzue **hutsetik**. Gainera, txantiloietan agertzen diren proba kasuak agertzeak ez du esan nahi proba guztiak daudenik. Zuek gehitu beharko dituzue **falta direnak**.
2. Laborategi honetan ez da informerik entregatu behar. eGelara jatorri fitxategiak (.adb eta .py) fitxategi batean konprimatuta (.zip) igo beharko dituzue. Gogoratu fitxategiaren orain arteko izendatze arauak errespetatzeaz (adibidez, Jsaez_Petxeberria_lab7.zip).
3. Ariketak zuzenak direla aurresuposatzen da, hau da, konpilazio errorerik ez dutela eta ondo funtzionatzen dutela. Horrela, ariketa ondo egoteak ez du punturik ematen, baina akatsak egoteak penalizatzen du. Behin soluzioa zuzena dela, honakoa da ebaluatuko dena (eta puntuatuko duena):
   a) **Proba kasuak**: Proba kasu guztiak gehitu dira, orokorretatik kritikoenetara? Batzuk ematen dira, baina beste asko falta dira.
   b) **Eraginkortasuna**: Beharrezkoak direnean “salatariak” erabiltzen dira begizta osoa ez korritzeko? Beharrezkoak ez diren baldintzak daude? Beharrezko parametro eta aldagaiak BAKARRIK definitzen dira?
   c) **Argitasuna**: Kodea tabulatua dago? Aldagaien izenek kodea ulertzen laguntzen dute? Funtzioaren bukaeran return BAKARRA dago? ....

1. ariketa: Errepikatuak ezabatu

Ariketa hau **ADAz** bakarrik ebatzi behar da.
Fitxategia (ADA): def_datuak.ads eta idatzi_zerrenda.adb (ez dira moldatu behar).
Txantiloiak (ADA): badago.adb, errepikatuak_ezabatu.adb eta proba_errepikatuak_ezabatu.adb

Azpiprograma bat inplementa ezazu, zenbaki osoen zerrenda batetik, elementu errepikaturik ez duen zerrenda baliokide bat sor dezan. Adibidez:

Sarrera:

1	2	1	1	3	3

Irteera:

1	2	3			

2. ariketa: Palindromoak (ERREKURTSIBO)

Ariketa hau **ADAz** bakarrik ebatzi behar da.
Fitxategia (ADA): datuak.ads (ez da moldatu behar).
Txantiloiak (ADA): palindromoa_da.adb, palindromoa_da_errekurtsiboa.adb eta proba_palindromoa_da.adb

Azpiprograma bat diseinatu eta inplementatu, letra xeheak eta zuriuneak baino ez dituen karaktere-zerrenda batetik, zerrenda horrek palindromo bat duen ala ez adierazten duen booleano bat itzultzen duena.

Palindromoa aurretik atzera eta atzetik aurrera berdin irakurtzen den hitz, zenbaki edo esaldi bat da; adibidez, “amama” edo “Iker, ireki!”.

Ariketa hori, ezinbestean, estrategia errekurtsibo baten bidez ebatzi behar da. Horretarako, palindromoa_da funtzioak (parametro gisa karaktere-zerrenda jasotzen duenak) palindromoa_da_errekurtsiboa azpiprograma (parametro gehigarriren bat jasoko duena) errekurtsiboki erabiliko du katea palindromoa den ala ez jakiteko.

3. ariketa: Osoko handien kenketa

Ariketa hau **Python**ez bakarrik ebatzi behar da.
Txantiloia: kenketa.py

Zenbaki handiak digituz digitu adierazteko 100 tamainako bektoreak erabiliko dira. Adibidez 1223334445678556111 zenbakia honela adieraziko litzateke:

0	1	2	...	88	89	90	91	92	93	94	95	96	97	98	99
0	0	0	...	4	4	5	6	7	8	5	5	6	1	1	1

Inplementa ezazu bi zenbakien arteko kenketa (digituz digitu) burutuko duen azpiprograma bat eta honi deituko dion programa nagusia, proba kasuekin.

4. ariketa: Konpilatzailea

Ariketa hau **ADAz** bakarrik ebatzi behar da.
Fitxategia (ADA): datuak_4.ads (ez da moldatu behar).
Txantiloiak (ADA): orekatu.adb eta proba_orekatu.adb

Konpilazio errore ohiko bat, programa batean parentesiak edo giltzak irekitzea eta ondoren ixtea ahaztea da, edo ireki aurretik ixtea. Garatu ezazu azpiprograma bat, zeinak, errore horiek topatuko dituen.

Adibidez:

```
if (a > b) then { b := a+n ) ; z := x+y }
                                ↑
                                Errorea!!!
```

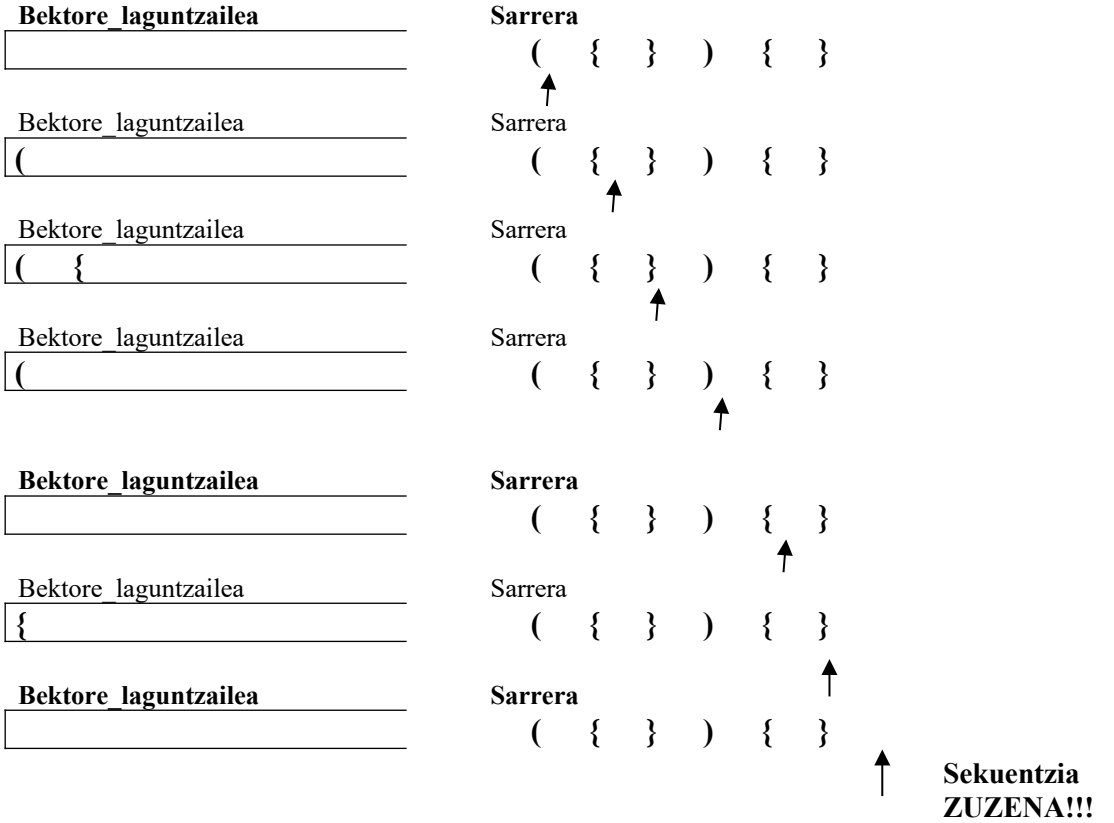
Jarraitu beharreko prozesuan bektore laguntzaile bat izango dugu eta sarrerako sekuentzia karakterez karaktere korritzea izango da:

- Parentesiak edo giltzak ez diren hizkiak ez dira tratatuko.
- Irekitzeko parentesi bat edo giltza bat aurkitzean, bektore laguntzailean gordeko da.
- Ixteko parentesi edo giltza bat aurkitzen bada, orduan azkenik gorde den elementuarekin bat egin beharko du. Kasu honetan elementua bektoretik ezabatuko dugu.

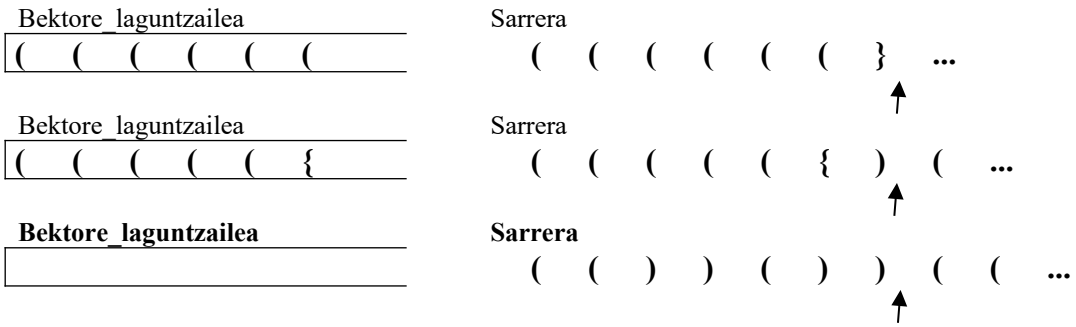
Adibide gisa hainbat kasu:

Adibide zuzena:

if(a= {b*c }) then {a:=c }



Adibide oker batzuk:



Okerra izango dira ere, sekuentziaren korritzea bukatzen denean bektore laguntzailean elementuren bat gelditzen denean.

5. ariketa: Anplitudeak

Ariketa hau **Pythonez** bakarrik ebatzi behar da.
Txantiloiak: *kalkulatu_anplitude_max.py*

Anplitude maximoa kalkulatzen duen azpiprograma idatz ezazu, zeinak osokoen zerrenda bat sarrera gisa jasotzen duen. Zerrenda hori goranzko zein beheranzko ordena jarraitzen duten azpi-zerrendez osatuta dago, zeintzuek anplitudeak irudikatzen dituzten. Segmentu bat goranzko ordenako azpi-zerrenda batez eta jarraian beheranzko ordenako beste azpi-zerrenda batez osatuta dago. Azpiprogramak segmentu luzeenaren (anplitude handienekoa) luzera itzuliko du.

Adibide gisa:

Ondorengo lista izanda, azpiprogramaren emaitza 10 izango da, segmentu luzeena baita (7tik 16 posiziora arte). Beste segmentua txikiagoa da (1etik 7ra). Ohartu zaitez, badagoela goranzko ordenan dagoen azpi-zerrenda bat baina ez den segmentua, ez baitu beheranzko azpi-zerrendarik.