INFORMATIKAREN OINARRIAK IKASGAIAREN AZTERKETA
2016ko ekainaren 28an

1. ARIKETA: POSTA-HELBIDE ELEKTRONIKOAK (2.75 + 0.5 puntu)

Ariketa honen asmoa da posta-helbide elektroniko multzo bat baliagarria den ala ez aztertzea. Posta-helbide elektroniko batek bi atal ditu, *a bildu* edo *arroba* batez banatuak (adibidez, *j.bond007@ikasle.ehu.eus*). Arrobaren ezkerreko azpikatea erabiltzaileari dagokio (*j.bond007*); arrobaren eskuineko azpikatea, berriz, domeinuari (*ikasle.ehu.eus*). Gainera, helbide bat baliagarria izan dadin, honako **baldintza** hauek bete behar ditu:

- **1B – Karaktere baliagarriak**: Alfabeto ingelesaren letra xeheak (‘a’...’z’), zenbakiak (‘0’...’9’), arrobak (‘@’) eta puntuak (‘.’) baino ezin ditu izan helbideak.
- **2B – Puntuak eta arroba**: Arroba karaktere bakar bat izan behar du, nahitaez, helbideak. Gainera, ezin ditu bi puntu izan jarraian.
- **3B – Erabiltzailearen atala**: Erabiltzailearen atalak, gutxienez, karaktere bat izan behar du. Gainera, lehendabiziko karaktereak, nahitaez, letra bat izan behar du. Azken karakterea, halaber, ezin da puntu bat izan. Gainerako karaktereei dagokienez, letrak, puntuak edo digituak izan daitezke, baina lehen digituaren ondoren digituak soilik egon daitezke (hau da, digituak agertzen badira, atzealdean egon behar dute denak batera).
- **4B – Domeinuaren atala**: Atal horrek, gutxienez, lau karaktere izan behar ditu, eta letrak edo puntuak soilik izan daitezke. Nahitaezkoa du puntu bat edukitzea, baina puntu hori ezin da egon arrobaren ondoan. Bi punturen artean, gutxienez, bi karaktere jarri behar dira, eta azken puntuak, nahitaez, azken-hirugarren (*.es*) edo azken-laugarren lekuan (*.eus*) egon behar du.

Honako hau inplementatu behar da (gutxienez):

a) **(*% 100 egin behar dutenentzat*) egiaztatu_karaktere_baliagarriak** funtzioa: *helbidea* izeneko karaktere-katea emanda, *1* balioa itzuliko du 1B baldintzako irizpideak betetzen baditu, eta *0* balioa, betetzen ez baditu ***(0.25 puntu).***

b) **(*% 100 egin behar dutenentzat*) egiaztatu_puntuak_eta_arroba** funtzioa: *helbidea* izeneko karaktere-katea emanda, *1* balioa itzuliko du 2B baldintzako irizpideak betetzen baditu, eta *0* balioa, betetzen ez baditu ***(0.25 puntu).***

c) **erauzi_erabiltzailearen_azpikatea** funtzioa: bi karaktere-kate emanda (*helbidea* eta *erabiltzailea* izenekoak), *helbidea* karaktere-katetik erabiltzailearen atala erauzi behar du (lehen karaktereti arrobaraino korrituz) eta *erabiltzailea* katean kopiatu (arroba ez du kopiatu behar). Gogoan izan *helbidea*k arroba bakar bat duela ***(0.35 puntu).***

d) **erauzi_domeinuaren_azpikatea** funtzioa: bi karaktere-kate emanda (*helbidea* eta *domeinua* izenekoak), *helbidea* karaktere-katetik domeinuaren atala erauzi behar du (arrobatik azken karaktereraino korrituz) eta *domeinua* katean kopiatu (arroba ez du kopiatu behar). Gogoan izan *helbidea*k arroba bakar bat duela ***(0.4 puntu).***

OHARRA: c) eta d) ataletan, ez ahaztu kate-amaierako karakterea (‘\0’) eransten *erabiltzailea* eta *domeinua* kateei.

e) **egiaztatu_erabiltzailea** funtzioa: *erabiltzailea* karaktere-katea emanda, *1* balioa itzuliko du 3B baldintzako irizpideak betetzen badira, eta *0* balioa, betetzen ez badira ***(0.75 puntu).***

f) **egiaztatu_domeinua** funtzioa: *domeinua* karaktere-katea emanda, *1* balioa itzuliko du 4B baldintzako irizpideak betetzen badira, eta *0* balioa, betetzen ez badira ***(0.75 puntu).***

g) **Programa nagusia**: aurreko ataleko funtzioak modu zuzenean edo ez-zuzenean erabiliz, posta-helbide elektroniko bat eskatuko dio erabiltzaileari, *helbidea* katean gordeko du (256 karaktere izango ditu gehienez), eta mezu bat erakutsiko du pantailan esanez helbidea baliagarria den ala ez. Baliagarria ez bada, aurkitu duen lehen akatsa erakutsi beharko du

(ikus exekuzio-adibidea). Prozesua errepikatu egingo da, harik eta sartutako helbidea "@@" izan arte ***(0.5 puntu)***.

Exekuzio-adibidea (***letra etzan, beltz eta azpimarratuz*** daude erabiltzaileak teklatutik sartutako datuak):

Idatzi posta-helbide elektroniko bat: ***j_b#o#n#d@ikasle.EHU.eus***
Helbide ez-zuzena: baliagarriak ez diren karaktereak ditu.
Idatzi posta-helbide elektroniko bat: ***jbond@ehu.es***
Helbide zuzena.
Idatzi posta-helbide elektroniko bat: ***007@ehu.eus***
Helbide ez-zuzena: erabiltzailearen atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***@ehu.eus***
Helbide ez-zuzena: erabiltzailearen atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***j007bond@ikasle***
Helbide ez-zuzena: erabiltzailearen atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***j.bond007@ehu.es***
Helbide zuzena.
Idatzi posta-helbide elektroniko bat: ***j..bond@ikasle.ehu.eus***
Helbide ez-zuzena: puntuen eta arrobaren baldintza ez du betetzen.
Idatzi posta-helbide elektroniko bat: ***j.bond@ikasle@ehu.eus***
Helbide ez-zuzena: puntuen eta arrobaren baldintza ez du betetzen.
Idatzi posta-helbide elektroniko bat: ***jbond@ehu***
Helbide ez-zuzena: domeinuaren atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***jbond@ikasle007.eus***
Helbide ez-zuzena: domeinuaren atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***jbond@.eus***
Helbide ez-zuzena: domeinuaren atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***ikasle@ehu.es***
Helbide zuzena.
Idatzi posta-helbide elektroniko bat: ***jbond@ikasle.a.eus***
Helbide ez-zuzena: domeinuaren atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***jbond@eus.ehu.ikasle***
Helbide ez-zuzena: domeinuaren atala ez da zuzena.
Idatzi posta-helbide elektroniko bat: ***@@***

ADI! Atal horiek %100eko azterketa egin behar dutenek soilik inplementatu behar dituzte. Gainerakoek funtzio horiek erabiltzeko aukera dute, nahiz eta inplementatu ez.

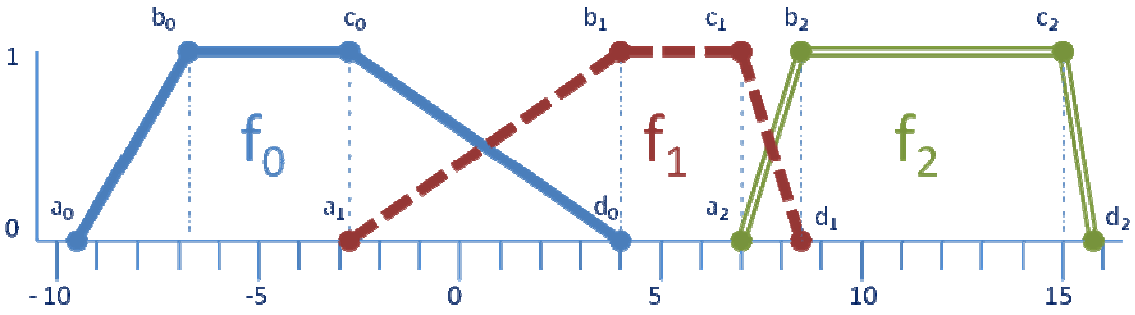
2. ARITEKA: MULTZO LAUSOAK (2.25 + 0.5 puntu)

Multzo lausoak oso erabiliak dira adimen artifizialaren arloan; adibidez, erabakiak hartzen laguntzeko sistemetan, ezagutza errepresentatzeko. Multzo lauso bakoitzak pertenentzia-funtzio bat du loturik (f: $\mathbb{R} \rightarrow [0,1]$). Ariketa honetan, joko dugu funtzio horiek trapezoidalak direla; hala, funtzio bakoitza zehazteko, *float* motako 4 balio ($a < b < c < d$) erabiliko dira:

- *a* beheko muga da. Funtzioak, puntu horretara arte, *0* balio du, eta, hortik aurrera, handiagotzen hasten da.
- *b* beheko oinarri-puntua da. Puntu horretatik aurrera, funtzioa ez da gehiago handiagotzen. Funtzioak *1* balioa hartzen du lehen aldiz, eta balio horri eusten dio puntu horretatik aurrera ere.
- *c* goiko oinarri-puntua da. Handik aurrera, funtzioa txikiagotzen hasten da berriro.
- *d* goiko muga da. Puntu horretatik aurrera, funtzioa ez da gehiago txikiagotzen; alegia, handik aurrera *0* da funtzioaren balioa.

Adibide honetan, 3 multzo lauso hautatu dira, honako pertenentzia-funtzioak (trapezioen mugak) dituztenak:

- f_0: $\{a_0, b_0, c_0, d_0,\} = \{-9.6, -6.9, -2.8, 4.0\}$
- f_1: $\{a_1, b_1, c_1, d_1,\} = \{-2.8, 4.0, 7.0, 8.5\}$
- f_2: $\{a_2, b_2, c_2, d_2,\} = \{7.0, 8.5, 15.0, 15.95\}$



Irudian ageri denez, trapezioak simetrikoak dira elkarrekiko; hau da, trapezio-pare bakoitzerako, baldintza hau betetzen da: ezkerreko trapezioaren *c* eta eskuineko trapezioaren *a* puntuak balio bera dute, eta ezkerreko trapezioaren *d* eta eskuineko trapezioaren *b* puntuak balio bera dute.

Array bat erabiliko da zenbait multzo lausoren pertenentzia-funtzioak zehazten dituzten mugak gordetzeko. Adibidez, aurreko adibiderako:

mugak

-9.6	-6.9	-2.8	4	-2.8	4	7	8.5	7	8.5	15	15.95
0	1	2	3	4	5	6	7	8	9	10	11

f_0 f_1 f_2

GARRANTZITSUA: Adibidea 3 trapeziotarako planteatuta badago ere, edozein trapezio kopurutarako balio behar du eskatzen den programak; horretarako, konstante bat definitu beharko litzateke (#define TRAPEZIO_KOPURUA xxx), eta *array*a eta kodea konstante horren arabera idatzi beharko lirateke.

Honako hau inplementatu behar da:

a) irakurri_datuak funtzioa: multzo lausoen pertenentzia-funtzioen mugak sartzeko eskatuko dio erabiltzaileari, eta array batean gordeko ditu mugok. Ez du inolako egiaztapenik egingo ***(0.2 puntu).***

b) egiaztatu_datuak funtzioa: multzo lausoen pertenentzia-funtzioen mugak dauzkan array bat emanda, 1 balioa itzuliko du muga horiek zuzenak baldin badira; 0 balioa, berriz, okerrak baldin badira. Kontuan izan ezen, mugak zuzenak izateko, pertenentzia-funtzio bakoitzeko laukoteak $a < b < c < d$ bete behar duela eta, gainera, trapezioek elkarrekiko simetrikoak izan behar dutela (alegia, trapezio-pare bakoitzerako: ezkerreko trapezioaren *c* eta eskuineko trapezioaren *a* puntuek balio bera izatea, eta ezkerreko trapezioaren *d* eta eskuineko trapezioaren *b* puntuek balio bera izatea; ertzetan ez dago betekizunik) ***(0.4 puntu).***

c) segmentuen_batezbestekoa funtzioa: multzo lausoen pertenentzia-funtzioen mugak dauzkan array bat emanda, *segmentu* guztien luzeren batezbesteko luzera itzuliko du ***(0.4 puntu).***

OHARRA: Multzo lauso baten *segmentua* da pertenentzia-funtzioak *1* balioa duen zati edo eremua. Hortaz, {*a*, *b*, *c*, *d*} mugen bidez zehaztutako multzo lauso baten *segmentua*ren luzera honela kalkulatzen da: $c - b$.

d) batazbestekoa_baino_handiagoa_den_lehen_trapezioa funtzioa: multzo lausoen pertenentzia-funtzioen mugak dauzkan array bat emanda, *segmentu* guztien luzeren batezbestekoa baino *segmentu* luzeagoa duen lehen multzo lausoaren posizioa itzuliko du. Halako multzo lausorik existituko ez balitz (hau da, *segmentu* guztiek luzera izanez gero), *-1* balioa itzuli beharko luke ***(0.75 puntu).***

e) (***% 100 egin behar dutenentzat soilik***) **erakutsi_segmentuaren_luzera** funtzioa: multzo lauso baten posizioa (0 eta TRAPEZIO_KOPURUA bitartekoa) eta multzo lausoen pertenentzia-funtzioen mugak dauzkan array bat emanda, adierazitako posizioko multzo lausoaren *segmentuaren* luzera kalkulatu eta pantailaratuko du ***(0.2 puntu).***

f) (***% 100 egin behar dutenentzat soilik***) **erakutsi_segmentuen_luzerak** funtzioa: multzo lausoen pertenentzia-funtzioen mugak dauzkan array bat emanda, aurreko funtzioa erabiliko du multzo lauso guztien *segmentuen* luzerak pantailaratzeko ***(0.3 puntu).***

g) **Programa nagusia**: aurreko ataletan deskribatutako funtzioak era zuzenean edo ez-zuzenean erabiliz, multzo lausoen pertenentzia-funtzioen mugak dauzkan arraya hasieratuko du, eta sartutako hasierako balio horiek zuzenak direla egiaztatuko du. Zuzenak ez badira, muga guztiak berriro sartzeko eskatuko dio erabiltzaileari, behar beste aldiz, muga guztiak zuzenak izan arte. Ondoren, multzo lausoen *segmentuen* luzerak erakutsiko ditu pantailan; eta, azkenik, *segmentu* guztien luzeren batezbestekoa baino *segmentu* luzeagoa duen lehen multzo lausoaren posizioa pantailaratuko du. Horrelako segmenturik ez balego (hau da, segmentu guztiak berdinak balira), hori adierazten duen mezu bat erakutsiko du pantailan ***(0.5 puntu).***

Exekuzio-adibidea (***etzan, beltz eta azpimarratuz*** daude erabiltzaileak teklatutik sartutako datuak):

```
Idatzi 0. trapezioaren balioak:   1.5   -2.5   3.5   4.5
Idatzi 1. trapezioaren balioak:   300   200   150   100
Idatzi 2. trapezioaren balioak:   7.8   9.6   1.7   -5. 8
Sartutako datuak ez dira zuzenak!
Idatzi 0. trapezioaren balioak:   -9.6   -6.9   -2.8   4
Idatzi 1. trapezioaren balioak:   -2.8   4   7   8.5
Idatzi 2. trapezioaren balioak:   7   8.5   15   15.95
  Hau da 0. trapezioaren segmentuaren luzera: 4.10
  Hau da 1. trapezioaren segmentuaren luzera: 3.00
  Hau da 2. trapezioaren segmentuaren luzera: 6.50
Batezbestekoa baino segmentu luzeagoa duen lehen trapezioa: 2
--------------------------------
Idatzi 0. trapezioaren balioak:   1   2   3   4
Idatzi 1. trapezioaren balioak:   3   4   5   6
Idatzi 2. trapezioaren balioak:   5   6   7   8
  Hau da 0. trapezioaren segmentuaren luzera: 1.00
  Hau da 1. trapezioaren segmentuaren luzera: 1.00
  Hau da 2. trapezioaren segmentuaren luzera: 1.00
Multzo lauso batek ere ez dauka segmentuen batezbesteko luzera baino segmentu luzeagorik.
```

ADI! Atal horiek %100eko azterketa egin behar dutenek soilik inplementatu behar dituzte. Gainerakoek funtzio horiek erabiltzeko aukera dute, nahiz eta inplementatu ez.

3. ARIKETA:FITXATEGIEN ARIKETA (3 PUNTU)

Gure ordenagailuko lan-direktorioan, multzo lauso ugari dauzkagu, zenbait fitxategitan gordeta. Aurreko ataleko **erakutsi_segmentuen_luzerak** funtzioa erabili nahi da (ez da berriro egin behar, aurreko ariketakoa berrerabili baizik), multzo horietariko bakoitzaren ezaugarriak aztertzeko.

Idatzi behar den programaren zeregina hau da: fitxategi baten izena eskatu behar du, izen horri "*.dif*" luzapena gehitu behar dio (MULTZO_LAUSOEN_LUZAPENA aldagaiean zehaztuta egon behar du luzapenak, aldatzeko aukera egon dadin), fitxategi hori irakurri behar du (adz.: "*trapezioak.dif*") eta array batean gordeko ditu hango datuak. Gainera, **erakutsi_segmentuen_luzerak** funtzioaz, arrayan gordetako multzoen ezaugarriak pantailaratu behar ditu. Programak behin eta berriro egingo du prozesu hori, harik eta erabiltzaileak, fitxategiaren izen gisa, puntu bat ('.') sartzen duen arte.

Honako hau inplementatu behar da:

a) irakurri_fitxategi_izena funtzioa: sarrerako argumentu gisa *fitxategi_izena* izeneko karaktere-array huts bat hartuko du, erabiltzaileari fitxategi izen bat (luzapenik gabe) idazteko eskatuko dio, izen hori *fitxategi_izena* karaktere-arrayan gordeko du, eta "*.dif*" luzapena erantsiko dio. Funtzio horrek ez du inolako egiaztapenik egin behar ***(0.25 puntu).***

b) irakurri_fitxategiko_datuak funtzioa: sarrerako argumentu gisa *mugak* izeneko array bat izango du multzo lausoen mugak gordetzeko; baita *izena* izeneko karaktere-array bat ere, izen hori duen fitxategiko datuak irakurtzeko. Lehendabizi, fitxategia irakurtze-moduan irekitzen saiatuko da funtzioa. Fitxategia behar bezala irekitzen bada, hura irakurri eta datuak *mugak* arrayan sartuko dira; kasu horretan, *1* balioa itzuliko du funtzioak. Fitxategia ireki ezin izan bada (existitzen ez delako, esate baterako), *mugak* arraya bere horretan utziko da, eta *0* balioa itzuliko du funtzioak ***(1.5 puntu).***

Hona hemen, adibide gisa, "trapezioak.dif" fitxategiaren edukia nolakoa den:

```
-9.600000 -6.900000 -2.800000  4.000000
-2.800000  4.000000  7.000000  8.500000
 7.000000  8.500000 15.000000 15.950000
```

Ikus daitekeenez, lerro bat erabiltzen da multzo lauso bakoitzaren trapeziorako. Joko dugu fitxategi guztien formatua zuzena dela.

c) Programa nagusia: aurreko ataletan deskribatutako funtzioak era zuzenean edo ez-zuzenean erabiliz, erabiltzaileari fitxategi-izen bat eskatuko dio (multzo difusoen datuak dituen fitxategiarena), irakurri egingo du, eta irakurritako datuak *mugak* izeneko arrayan gordeko ditu, **irakurri_fitxategiko_datuak** funtzioaren bidez. Ondoren, *mugak* arraya **erakutsi_segmentuen_luzerak** funtzioari pasatuko zaio, multzo lausoen ezaugarriak pantailan ager daitezen ***(1.25 puntu).***

Programak behin eta berriro eskatuko du fitxategi-izen bat, harik eta erabiltzaileak puntu bat ('.') sartzen duen arte. Erabiltzaileak sartutako fitxategi-izeneko fitxategirik existitzen ez bada, mezu bat agerraraziko du pantailan.

Exekuzio-adibidea (***etzan, beltz eta azpimarratuz*** daude erabiltzaileak teklatutik sartutako datuak):

```
Idatz ezazu fitxategiaren izena (luzapenik gabe; irteteko, sartu puntu bat): alfa
  Fitxategi hori ez da existitzen!
Idatz ezazu fitxategiaren izena (luzapenik gabe; irteteko, sartu puntu bat): trapezioak.txt
  Fitxategi hori ez da existitzen!
Idatz ezazu fitxategiaren izena (luzapenik gabe; irteteko, sartu puntu bat): trapezioak
0. trapezioaren balioak irakurtzen.
1. trapezioaren balioak irakurtzen.
2. trapezioaren balioak irakurtzen.
  Hau da 0. trapezioaren segmentuaren luzera: 4.10
  Hau da 1. trapezioaren segmentuaren luzera: 3.00
  Hau da 2. trapezioaren segmentuaren luzera: 6.50
Idatz ezazu fitxategiaren izena (luzapenik gabe; irteteko, sartu puntu bat): .
Programaren amaiera.
```