3.2 Gaia. Azpiprogramen ariketak

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea

1. Ariketa (ADA)

- Azpiprograma bat egin, erabiltzaileari zenbaki bat eskatzen diona eta zenbaki hori positiboa dela ziurtatzen duena. Ez bada, berriz eskatuko lioke, behin eta berriz, positiboa izan arte
  - Espezifikazioa

Sarrera: -
Aurre: -
Irteera: zenbaki osoko bat: zen
Post: zen: balioa1 >0

Soluzioa

```
with Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer_Text_Io;
procedure positiboa_eskatu(Zen: out Integer) is
begin
   loop
   put_line("0 baino handiagoko osoko bat sartu: ");
   get(Zen);
   if (Zen<=0) then
         put_line("Sartutako balioa ez da egokia");
      end if;
      exit when Zen >0;
   end loop;
end positiboa_eskatu;
```

Programa nagusia

```
with positiboa_eskatu, Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer _Text_Io;

procedure positibo_nagusia is
   Zen: Integer;
begin
   put_line("Azpiprogramari deituko diogu...");
   positiboa_eskatu(Zen);
   new_line;
   put("Azpiprogramak itzulitako azpiprograma: ");
   put(Zen);
   new_line;
end positibo_nagusia;
```

2. Ariketa (ADA)

- Azpiprograma bat egin, zenbaki bat eta posizio bat emanda, posizio horretako digitua itzuliko duena
  - Espezifikazioa:

Sarrera: Bi zenbaki osoko: zen eta pos
Aurre: zen> 0 eta pos>=1 eta zen ez du 0 digiturik.
Irteera: zenbaki osoko bat: dig
Post: 0<=dig<=9, non dig zen zenbakiaren pos posizioan dagoen digitua da, eskuinetik hasita.

Soluzioa

```
function digitua_posizioan (Zen: Integer; pos:
Integer) return Integer is
   Auxiliarra: Integer := Zen;
   Kontagailua: Integer := 1;
begin
   while (Kontagailua<pos) loop
      Auxiliarra:=Auxiliarra/10;
      Kontagailua:=Kontagailua+1;
   end loop;
   return(Auxiliarra rem 10);
end digitua_posizioan;
```

Proben programa

```
with digitua_posizioan, Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer_Text_Io;

procedure probak_digitua_posizioan is
   Zenbakia, Emaitza, Posizioa: Integer;
begin
   Zenbakia:= 1234;
   Posizioa:=2;
   Emaitza:= digitua_posizioan(Zenbakia, Posizioa);
   if (Emaitza = 0) then
      put("Posizioa digitu kopurua baina handiagoa da");
   else
      put(Zenbakia);
      put(" ren ");
      put(Posizioa);
      put(" posizioan dagoen digitua ");
      put(Emaitza);
      put(" da.");
   end if;
   new_line;

   -- Kasu gehiago frogatu beharko lirateke
end probak_digitua_posizioan;
```

3. Ariketa

- Azpiprograma bat egin erabiltzaileari bi zenbaki osoko eskatzen dizkiona eta horien seinua aldatzen duena
  - Espezifikazioa

Sarrera: 2 zenbaki osoko: zen1 eta zen2
Aurre: num1: val1 y num2: val2
Irteera: 2 zenbaki osoko: zen1 eta zen2
Post: zen1: bal1*-1 y zen2: bal2*-1

1. bertsioa

```
with Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer_Text_Io;

procedure seinu_aldaketa_nagusia_v1 is
   Zen1, Zen2: Integer;
   procedure seinu_aldaketa(Zen1, Zen2: out Integer) is begin
      put_line("Sartu lehenengo zenbakia: ");
      get(Zen1);
      put_line("Sartu bigarren zenbakia: ");
      get(Zen2);
      put_line ("Sartutako zenbakiak: ");
      put(Zen1);
      put(Zen2);
      Zen1:= Zen1*(-1);
      Zen2:= Zen2*(-1);
   end seinu_aldaketa;
   begin  -- seinu_aldaketa_nagusia_v1 progrma hasten da
      seinu_aldaketa(Zen1,Zen2);
      put_line ("Seinua aldatu ostean, zenbakiak dira: ");
      put(Zen1);
      put(Zen2);
end seinu_aldaketa_nagusia_v1;
```

2. bertsioa

```
with Ada.Text_Io, Ada.Integer_Text_Io;
use Ada.Text_Io, Ada.Integer_Text_Io;

procedure seinu_aldaketa_nagusia_v2 is
   Zen1, Zen2: Integer;
   procedure seinu_aldaketa(Zen1, Zen2: in out Integer) is begin
      Zen1:= Zen1*(-1);
      Zen2:= Zen2*(-1);
   end seinu_aldaketa;
   begin  -- seinu_aldaketa_nagusia_v1 progrma hasten da
      put_line("Sartu lehenengo zenbakia: ");
      get(Zen1);
      put_line("Sartu bigarren zenbakia: ");
      get(Zen2);
      put_line ("Sartutako zenbakiak: ");
      put(Zen1);
      put(Zen2);
      seinu_aldaketa(Zen1,Zen2);
      put_line ("Seinua aldatu ostean, zenbakiak dira: ");
      put(Zen1);
      put(Zen2);
end seinu_aldaketa_nagusia_v2;
```

4. Ariketa (ADA eta Python)

- Azpiprograma bat egin, zeinak 2. mailako ekuazio baten bi erroen balioa itzultzen duen (ax2 + bx + c = 0), a, b eta c balioak emanda
  - Espezifikazioa

Sarrera: 3 zenbaki erreal: a, b eta c
Aurre: a-ren balioa ez da 0
Irteera: 2 zenbaki erreal: x1 eta x2
Post: x1: (-b + sqrt(b2 – 4ac))/2a, x2: (-b - sqrt(b2 - 4ac))/2a

ADA

```
with Ada.Text_Io, Ada.Float_Text_Io, Ada.Numerics.Elementary_Functions;
use Ada.Text_Io, Ada.Float_Text_Io, Ada.Numerics.Elementary_Functions;

procedure ekuazioa_2gradu_nagusia is
   A, B, C, R1, R2 : Float;
   procedure ekuazioa_2gradu(A, B, C: Float; X1,X2: out Float) is begin
      X1:= (-B + sqrt(B**2-4*A*C))/(2*A);
      X2:= (-B - sqrt(B**2-4*A*C))/(2*A);
   end seinu_aldaketa;
   begin
      put_line("Sartu a-ren balioa: ");
      get(A);
      put_line("Sartu b-ren balioa: ");
      get(B);
      put_line ("Sartu c-ren balioa: ");
      get(C);
      ekuazioa_2gradu(A,B,C,R1,R2);
      put_line ("Erroak dira: ");
      put(R1);
      put(R2);
end ekuazioa_2gradu_nagusia;
```

Python

```
from math import sqrt

def ekuazioa_2gradu(a, b, c):
    X1 = (-b + sqrt(b**2-4*a*c))/(2*a)
    X2 = (-b - sqrt(b**2-4*a*c))/(2*a)

def ekuazioa_2gradu_nagusia is
      a = float(input("Sartu a-ren balioa: "))
      b = float(input("Sartu b-ren balioa: "))
      c = float(input("Sartu c-ren balioa: "))
      r1,r2 = ekuazioa_2gradu(a,b,c);
      print("Erroak dira: "+ str(r1) + "eta" + str(r2))
end ekuazioa_2gradu_nagusia;
```

Proposatutako ariketa

- *n* zenbaki bikoita emanda (2 baino handiagoa), *n* batura ematen duten bi zenbaki lehenak (faktore lehenak deiturikoak) eman
  - Adibidez:
    - 4: 2 + 2
    - 6: 3 + 3
    - 8: 3 + 5
    - 22: 3 + 19, 5 + 17 eta 11+11
    - 24: 5 + 19, 7 + 17 eta 11+13
    - ...

Txantiloiak (ADA)

- Azpiprograma hau ebazteko beharrezko txantiloiak eGelan daude, faktore_lehenak.zip fitxategian
  - probak_faktore_lehenak_kalkulatu.adb
  - faktore_lehenak_kalkulatu.adb
  - lehenengo_faktore_lehenak_kalkulatu.adb
  - hurrengo_faktore_lehenak_kalkulatu.adb
  - hurrengo_zenbaki_lehena.adb

Proba-programa

- probak_faktore_lehenak_kalkulatu.adb
  - Lehenengo bi proba kasuak ematen dira, baina gehiago gehitu behar dira *faktore_lehenak_kalkulatu* ondo dabilela ziurtatzeko

Algoritmoa

- faktore_lehenak_kalkulatu.adb
  - existitzen den lehenengo faktore lehenenen bikotea jasotzen da
    - lehenengo_faktore_lehenak_kalkulatu.adb
  - Ostean, begiztan sartzen da. Iterazio bakoitzean existitzen den hurrengo faktore lehenen bikotea jasotzen da
    - hurrengo_faktore_lehenak_kalkulatu.adb

Azpiprograma laguntzaileak

- *lehenengo_faktore_lehenak_kalkulatu* eta *hurrengo_faktore_lehenak_kalkulatu* azpiprogramek honakoak erabiliko dituzte:
  - lehena_da, zeinak zenbaki bat emanda, lehena den edo ez kalkulatzen duen
  - hurrengo_zenbaki_lehena, zeinak zenbaki lehen bat emanda, hurrengo zenbaki lehena itzuliko duen