1. **<u>ariketa</u>.** Ondoko espezifikazioa betetzen duen funtzioa egin nahi da:

```
N: constant Integer := 4;
type Hitza is array(1 .. N) of Character;

function Anagrama_Da(X, Y: in Hitza) return Boolean;
-- Aurrebaldintza: X eta Y-k letra maiuskulak dituzte
-- Postbaldintza: emaitza True izango da Y X-eko letren ordena
--        aldatuz lor badaiteke (permutazioa da), eta False bestela
```

1. algoritmoa

```
      YKopia: Hitza := Y;
      j, Aurkituak: Integer;
    begi
      Aurkituak := 0;
      for i in 1 .. N loop
        j := 1;

        while (X(i) /= YKopia(j)) and (j < N) loop
          j := j + 1;
        end loop;

        if X(i) = YKopia(j)
        then YKopia(j) := ' '; -- ezabatu, hurrengoan ez aurkitzeko
             Aurkituak:= Aurkituak + 1;
        end if;
      end loop;

      return Aurkituak = N;
    end Anagrama_Da;
```

2. algoritmoa

```
      Alfabetoa: array('A'..'Z') of Integer;
        -- letra bakoitzeko, bere agerpen-kopurua kontatuko da
      j: Integer;
begin
  for l in 'A' .. 'Z' loop Alfabetoa(l):= 0; end loop;

  for i in 1 .. N loop Alfabetoa(X(i)) := Alfabetoa(X(i)) + 1; end loop;

  for i in 1 .. N loop Alfabetoa(Y(i)) := Alfabetoa(Y(i)) - 1; end loop;

  j := 'A';
  while (Alfabetoa(j) = 0) and (j < 'Z') loop
    j := Character'Succ(j);
  end loop;

  if Alfabetoa(j) = 0 then return True; else return False; end if;
end Anagrama_Da;
```

Ariketak:

1.1 Saiatu ulertzen algoritmo bakoitza.

1.2 Kalkulatu algoritmo bakoitzaren ordena, hitzaren tamainaren arabera. Zein da eraginkorrena?

2. **<u>ariketa.</u>** Kalkulatu algoritmo bakoitzaren ordena, matrizearen tamainaren arabera. Zein da eraginkorrena?

```
type Matrizea is array (1 .. Lerroak, 1 .. Zutabeak) of Integer;
```

```
function Batura1 (Mat :
                  in Matrizea)
                return Integer is
       Guztira : Integer;
       I, J: Integer;
begin
   Guztira:= 0;
   I := 1;

   while (I <= Lerroak) loop
     J := 1;
     while (J <= Zutabeak) loop
       Guztira:=
              Guztira + Mat(I, J);
       J := J + 1;
     end loop;

     I := I + 1;
   end loop;

   return Guztira;
end Batura1;
```

```
function Batura2(Mat: in Matrizea
)
               return Integer is
       Guztira: Integer;
       I, J: Integer;
begin
   Guztira:= 0;
   I := 1; J := 1;

   while (I <= Lerroak) loop
     Guztira:=
            Guztira + Mat(I, J);
     if (J < Zutabeak)
     then J := J + 1;
     else J := 1;
          I := I + 1;
     end if;
   end loop;

   return Guztira;

end Batura2;
```

```
function Batura3(Mat :
                   in Matrizea )
                 return Integer is

   Guztira: Integer;

begin
  Guztira:= 0;

  for I in 1 .. Lerroak loop
    Guztira := Guztira +
      Errenkadaren_Batura (Mat, I);
  end loop;

  return Guztira;
end Batura3;
```

```
function Errenkadaren_Batura(
          Mat : in Matrizea;
          Errenkada: in Integer)
                    return Integer
is
   Guztira: Integer;
begin
 Guztira:= 0;
 for J in 1 .. Zutabeak loop
     Guztira:= Guztira +
               Mat(Errenkada, J);
 end loop;
 return Guztira;
end Errenkadaren_Batura;
```

3. **ariketa.** Honako algoritmoaren konplexutasun-kostua aztertu:

```
function Muga_Gainditzen (n: Integer) return Integer is

    Jauzia: Integer := 1;
    Posizioa: Integer := 0;
begin
    while Posizioa < n loop
          Posizioa := Jauzia;
          Jauzia := jauzia * 2;
    end loop;
    return Posizioa;
end Muga_Gainditzen;
```

4. **ariketa. ariketa**. Ikasleen zerrenda bat dugu, eta ikasle bakoitzeko matrikulatu den asignaturen zerrenda.

- Ikasle baten zenbakia emanda, zenbat denbora hartuko du ikasle hori algebran matrikulatuta dagoen jakiteak?

- Asignatura bakoitzeko, matrikulan dituen ikasleen zerrenda lortu nahi da. Zein da algoritmo horren kostua?

5. ariketa. Hirugarren kurtsoko ikasle batek algoritmo bat egin du, "primitiva"ko txartelen zerrenda emanda (bakoitza dagokion sariarekin) sari handienak dauzkaten 1.000 txartelak lortzeko.

```
type Txartelen_Zerrenda is record
    Taula: array (1 .. Max) of Txartela;-- Txartela mota definituta dago
    Zenbat: Integer;
end record;
```

```
function 1000_Hoberenak (Txartelak : in Txartelen_Zerrenda)
                                        return Txartelen_Zerrenda is
-- post: Txartelak zerrendako 1.000 hoberenak lortu dira emaitzan

   Emaitza: Txartelen_Zerrenda;
begin
   Hasieratu(Emaitza);
   I:= 1;

   while (I <= Txartelak.Zenbat) loop
       if Emaitza.Zenbat < 1000 or
          Saria(Txartelak.Taula(I)) > Saria(Txikiena(Emaitza))
       then sartu txartel hau Emaitza-ren bukaeran
            Ordenatu(Emaitza); -- burbuilaren bidez
       end if;
   end loop;

   return Emaitza;
end 1000_Hoberenak;
```

Ondokoa eskatzen da:

a) Aztertu soluzioa eta bere kostua kalkulatu

b) Algoritmo hori 14.000.000 txartelen zerrenda batean aplikatzen denean, ikasleak dio 8/9 ordu hartzen dituela, 1.000 txartel hoberenak lortzeko. Lagunduko zenioke algoritmoa hobetzen? Zure erantzuna arrazoitu.

6. ariketa Klase hau emanda:

```
public class Liburu {
 int mota;
   // 3 balio posible: 1 (abenturak), 2 (historia) edo 3 (zientzia)
 String izenburua;
 String egileak;
}
```

Liburuen array bat ordenatu nahi da, lehenengo abenturazkoak egoteko, ondoren historiakoak eta azkenean zientziakoak. Kategoria bakoitzaren barruan, ordenak ez du garrantzirik. 2 algoritmo eman dizkigute:

Algoritmo 1

```
public void bubbleSort(Liburu[] taula) {
     int out, in;
     for (out = taula.length - 1; out > 0; out--)
       for (in = 0; in < out; in++)
         if (taula[in].mota > taula[in + 1].mota)
           swap(taula, in, in + 1);
   }
```

Algoritmo 2

```
public void misterio(Liburu[] taula) {
     int in, i1, i2, i3 = 0;
     Liburu[] t1, t2, t3; // 3 array lagungarri
     t1 = new Liburu[taula.length];
     t2 = new Liburu[taula.length];
     t3 = new Liburu[taula.length];

     for (in = 0; in < taula.length; in++)
      if ( taula[in].mota == 1 ) { t1[i1] = taula[in]; i1++; }
      else if ( taula[in].mota == 2 ){ t2[i2] = taula[in]; i2++; }
      else if ( taula[in].mota == 3 ){ t3[i3] = taula[in]; i3++; }
     }
     int i = 0;
     for (in = 0; in < i1; in++) { taula[i] = t1[in]; i++};
     for (in = 0; in < i2; in++) { taula[i] = t2[in]; i++};
     for (in = 0; in < i3; in++) { taula[i] = t3[in]; i++};
   }
```

Hau eskatzen da:

· Azaldu zer egiten duen algoritmo bakoitzak

· Esan, modu arrazoituan, zein den algoritmo bakoitzaren kostua, eta horren arabera, 2 algoritmoetatik zein izango litzatekeen eraginkorrena

7. ariketa Bi algoritmo hauek emanda:

Algoritmo 1

```
public static int count(int[] a) {
  int N = a.length;
  int cnt = 0;
  for (int i = 0; i < N; i++) {
    for (int j = i+1; j < N; j++) {
      if (a[i] + a[j] == 0) {
        cnt++;
      }
    }
  }
  return cnt;
}
```

Algoritmo 2

```
public static int count(int[] a) {
  int N = a.length;
  Arrays.sort(a); // mergesort
  int cnt = 0;
  for (int i = 0; i < N; i++) {
    int j = Arrays.binarySearch(a, -a[i]); // bilaketa bitarra
    if (j > i) cnt++;
  }
  return cnt;
}

public int binarySearch(int[] a, int x)
// Aurrebaldintza:
// Postbaldintza: x duen elementuaren posizioa bueltatzen du
        0 bueltatuko da x ez badago a arrayan
```

Hu eskatzen da:

a) Esan zer egiten duen algoritmo bakoitzak

b) Beraien kostua kalkulatu era arrazoituan