AULA POLITÈCNICA
/ INFORMÀTICA

X. Franch - J. Marco - X. Molinero
J. Petit - F. Xhafa

Introducció a la programació

Problemes resolts

EDICIONS UPC

AULA POLITÈCNICA 125

Introducció a la programació

Problemes resolts

AULA POLITÈCNICA
/ INFORMÀTICA

X. Franch - J. Marco - X. Molinero
J. Petit - F. Xhafa

Introducció a la programació

Problemes resolts

EDICIONS UPC

Primera edició: setembre de 2006

Disseny de la coberta: Jordi Calvet

Edicions de la Universitat Politècnica de Catalunya, SL
Jordi Girona Salgado 31, 08034 Barcelona
Tel.: 934 016 883 Fax: 934 015 885
Edicions Virtuals: www.edicionsupc.es
E-mail: edicions-upc@upc.edu

ISBN: 978-84-9880-162-0

Pròleg

El nostre objectiu a l'escriure aquest llibre d'exercicis i problemes resolts ha estat confeccionar una obra que s'adaptés a les característiques de les assignatures d'introducció a la programació que impartim a la Facultat d'Informàtica de Barcelona i a la Facultat de Matemàtiques i Estadística de la Universitat Politècnica de Catalunya. Aquesta edició és l'evolució natural dels continguts de l'obra *Informàtica bàsica* dels mateixos autors. Per això, hem intentat, en la mesura dels possibles, cobrir els aspectes bàsics i fonamentals en un curs d'introducció a la programació, tant en el vessant teòric com en la seva aplicació. D'aquesta forma, volem solucionar parcialment l'absència de material específic per a aquestes assignatures.

El resultat ha estat un text organitzat en dues unitats diferenciades, una d'enunciats i una de solucions.

A la primera unitat, plantegem exercicis d'autoavaluació i enunciats de problemes. Els exercicis solen ser enunciats curts i concisos, cadascun dels quals involucra una tècnica o un coneixement específic del temari. En general, es tracta d'exercicis que il·lustren els conceptes que s'han introduït a les classes de teoria. Els problemes presenten una major complexitat que la dels exercicis i, freqüentment, involucren diferents conceptes teòrics que cal combinar per obtenir una bona solució. La majoria dels problemes han aparegut en exàmens parcials o finals recents en assignatures d'introducció a la programació de la UPC (tot i que han estat adaptats i unificats).

La segona unitat planteja la resolució dels problemes, intentant mitigar les dificultats que sovint tenen els alumnes en aplicar sobre casos concrets els conceptes que han adquirit i, fins i tot, entès en les classes de teoria. Per això, hem intentat comentar exhaustivament les solucions per facilitar l'aprenentatge dels diversos mètodes de resolució dels problemes.

El llibre utilitza un subconjunt bàsic del llenguatge de programació C++ per descriure els algorismes. Aquest subconjunt s'ha mantingut expressament tan petit com ha estat possible: només s'utilitza un paradigma de programació imperativa procedural i no es tenen en compte les capacitats d'orientació a objectes que ofereix aquest llenguatge (tot i que sí s'usen objectes per lectura, escriptura, cadenes de caràcters i taules).

Per fer funcionar aquests algorismes escrits en C++, el lector només haurà de reordenar els diferents fragments de cada programa i afegir-hi les capçaleres que, per raons de brevedat, no hem repetit a cada solució. En particular, només s'utilitzen unes poques funcionalitats de les llibreries `<iostream>`, `<string>`, `<vector>` i `<cmath>`.

Voldríem agrair als nostres companys del departament de Llenguatges i Sistemes Informàtics que han fet docència en aquestes assignatures la seva participació en la confecció d'alguns dels enunciats dels problemes que apareixen en aquests llibre, així com els seus comentaris sobre versions preliminars del material.

Esperem que el llibre sigui d'utilitat als estudiants de programació de la UPC i d'altres universitats de parla catalana.

Barcelona, juliol de 2006

Els autors

Índex

Part I

Exercicis i problemes

1

Introducció

Exercicis

1. Penseu en totes les activitats que vàreu fer ahir. En quines d'elles sou conscients d'haver usat un sistema informàtic? Quines eren les seves entrades? Quines eren les seves sortides? Quins errors es podien produir?

2. Expliqueu a un turista perdut com anar de la Plaça Major a l'Aeroport del Prat tot utilitzant transports públics en una dia de vaga de taxis.

3. Expliqueu com multiplicar dos nombres a un nen que sap sumar i coneix les taules de multiplicar.

4. Què és un algorisme?

5. Digueu quin és el contrari de les afirmacions següents:

   - a és més petit que b.
   - Plou i fa sol.
   - No plou i no fa sol.
   - Dues rectes tenen un punt en comú.
   - Tots els camins porten a Roma.
   - Tots els ànecs són blancs i volen.
   - P i Q.
   - P o Q.
   - P o Q però no les dues.

6. Feu les taules de veritat de **no** (a **i** b) i de **no** a **o no** b. Quina és la vostra conclusió? Feu el mateix per **no** (a **o** b) i per **no** a **i no** b.

7. Feu les taules de veritat de **no** (a **i** b **i** c) i de **no** a **o no** b **o no** c. Quina és la vostra conclusió? Feu el mateix per **no** (a **o** b **o** c) i per **no** a **i no** b **i no** c.

8. Epimènides de Creta va afirmar: "Tots els cretencs són uns mentiders". Què en penseu?

9. En el codi ASCII, la '`a`' es representa amb el valor numèric 97, la '`A`' amb el 65 i el '`0`', amb el 48. Digueu els valors numèrics que representen els caràcters '`3`', '`8`', '`M`' i '`Z`' en aquest codi.

10. Quins són els tipus bàsics de C++? Digueu quins valors són admissibles per a cada tipus. Feu una taula on per a cada operador llisteu quins són els tipus dels seus operands i quin és el tipus del seu resultat. Assenyaleu quins errors es poden donar en aplicar aquests operadors.

11. Digueu de quin tipus són les expressions següents i avalueu-les:

    - `12`
    - `12.0`
    - `3 + (4 * 5)`
    - `(3 + 4) * 5`
    - `3 + 4 * 5`
    - `((((3 + 4) * 5)))`
    - `8 / 4`
    - `8 / 3`
    - `8.0 / 3.0`
    - `5 / 0`
    - `2 == 3`
    - `2 >= 3`
    - `3.0 >= 2.9`
    - `(3 % 2 == 0) == (7 / 2 > 5)`
    - `4 / (8 / 10)`
    - `(4 / 8) / 10`
    - *true and false or true*
    - *true and* (*false or true*)
    - (*true and false*) *or true*
    - *not* (`'S' < 'D'`)
    - `(5 + 1) / (9 % 3)`
    - `"miquel"`
    - `"miquel > "isabel"`
    - `"hola" + " " + "miquel"`
    - `double(34)`
    - `int(4.9) + int(4.1)`
    - `int(4.9 + 4.1)`
    - `int('9')`
    - `char(65)`
    - `int('2') >= int('9')`
    - `char(65)`

12. Donades aquestes declaracions:

    ```
    int i, j, k;
    double x, y, z;
    char a, b, c;
    bool p, q, r;
    ```

    digueu quines de les expressions següents són inadmissibles en C++ i per què:

    - `a`
    - `i+3 <= x`
    - `(j-k) / i`

- `(j-k) / x`
- `(j-k) / int(x)`
- `(i * j - z >= x * y * y) == (not p and q)`
- `|x - y| + z`
- $\sqrt{\texttt{x}} > 0$
- $\texttt{x}^2 - \texttt{y}^2$
- `not (p and q and r and c>='A')`
- `(p + r) + i`
- `i <= j <= k`
- `p == q == r`
- `3 * i - (j*z) * x`
- `a >= b and b > c`
- `(3*x) / (3.1416*y))`
- $\frac{\texttt{x} + \texttt{y} + \texttt{z}}{\texttt{i}}$

13. Expliqueu com es comporten els operadors de divisió entera i mòdul (`/` i `%`) amb operands negatius. Calculeu `7 / 4`, `-7 / 3`, `7 / -3`, `-7 / -3`, `7 % 3`, `-7 % 3`, `7 % -3`, i `-7 % -3`.

14. Donades aquestes inicialitzacions:

```
int z = 0;
int u = 1;
double x = 3.5;
bool t = true;
```

avalueu les expressions següents i digueu quines poden produir problemes:

- `not (z <= 1 and x >= 0 and t)`
- `z == 0 and x <= 0 or u == 1 and x >= 0`
- `u/z > -2`
- `z != 0 and u/z > -2`
- `u/z > -2 and z != 0`
- `z == 0 or u/z > -2`
- `u/z > -2 or z == 0`
- `z == 0 or 24%z != 1`
- `z == 0 and 24%z != 1`
- `x == 3.5`

15. Digueu quina és la sortida de cadascun dels programes següents sense executar-los:

```
// Programa 1
int main () {
    int a = 21, b = 666;
    cout << a + b << endl;
}

// Programa 2
int main () {
    int a = 24, b = 32;
    cout << a << " + " << b << " = " << a + b << endl;
}
```

```
// Programa 3
int main () {
    int a = 12, b = 8;
    if (a >= b) cout << a << endl;
    else cout << b << endl;
}

// Programa 4
int main () {
    int n = 10;
    for (int i = 1; i <= n; ++i) {
        cout << i << endl;
    }
}

// Programa 5
int main () {
    int n = 10;
    for (int i = 0; i < n; ++i) {
        cout << i << endl;
    }
}

// Programa 6
int main () {
    int n = 10, i;
    for (i = 0; i < n; ++i) {
        cout << i << endl;
    }
    cout << i << endl;
}

// Programa 7
int main () {
    int n = 5;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            cout << i << ',' << j << endl;
        }
    }
}

// Programa 8
int main () {
    int n = 5;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < i; ++j) cout << i << ',' << j << endl;
    }
}
```

```
// Programa 9
int main () {
    for (char c = 'A'; c <= 'Z'; ++c) cout << c;
    cout << endl;
    for (char c = 'a'; c <= 'z'; ++c) cout << c;
    cout << endl;
    for (char c = '0'; c <= '9'; ++c) cout << c;
    cout << endl;
}
```

16. Per als problemes següents, identifiqueu quines són les entrades, quines són les sortides, quina relació hi ha entre les entrades i les sortides, quines condicions han de complir les entrades i quins errors es poden donar. No feu els algorismes:

    - Calcular la suma de dos nombres reals.
    - Calcular el producte de dos nombres reals.
    - Calcular el quocient de dos nombres reals.
    - Calcular el quocient i el residu de dos nombres enters.
    - Calcular el valor absolut d'un nombre real.
    - Calcular l'arrel quadrada d'un nombre real.
    - Resoldre una equació lineal.
    - Resoldre una equació de segon grau.
    - Trobar el valor més alt d'una llista d'enters.
    - Trobar la mitjana de les notes dels alumnes d'una classe.
    - Calcular la distància entre dos punts del pla.
    - Calcular la distància entre dos punts de l'espai.
    - Esbrinar si dues línies són iguals, paral·leles o s'intersecten.
    - Simplificar una fracció.
    - Decidir si dues fraccions representen el mateix nombre racional.

17. Dissenyeu un algorisme que llegeixi dos nombres reals i escrigui la seva suma.

18. Dissenyeu un algorisme que llegeixi un nombre enter i escrigui si és parell o senar.

19. Dissenyeu un algorisme que llegeixi un nombre enter entre 100 i 999 i escrigui si és cap-i-cua.

20. Dissenyeu un algorisme que, donats tres enters que representen hores, minuts i segons, doni l'equivalent en segons.

21. Dissenyeu un algorisme que, donats tres enters que representen hores, minuts i segons, sumi un segon i doni el resultat en el mateix format.

22. Dissenyeu un algorisme que, donada una quantitat de segons, digui quantes hores, minuts i segons representa.

23. Dissenyeu un algorisme que faci la conversió de dolàrs a euros.

24. Dissenyeu un algorisme que donada una quantitat en euros, doni el nombre mínim de bitllets i monedes necessaris, sabent que es disposa de monedes de 1, 2, 5, 10, 20 i 50 cèntims i 1 i 2 euros, i bitllets de 5, 10, 20, 50, 100, 200 i 500 euros.

25. Escriviu expressions que, donat un real r, calculin.

    - la part entera per defecte de r,
    - la part entera per excés de r,
    - el valor enter més proper a r.

26. Escriviu expressions booleanes que, donat un caràcter c, indiquin.

    - si c és una lletra minúscula,
    - si c és una lletra majúscula,
    - si c és un dígit,
    - si c no és ni minúscula ni majúscula ni dígit,
    - si c és una vocal,
    - si c és una consonant,
    - si c és un símbol de puntuació.

27. Escriviu una expressió entera que es correspongui al valor d'un dígit donat (caràcters de '`0`' a '`9`'). Per exemple, per al caràcter '`3`', cal donar l'enter 3.

28. Escriviu una expressió que converteixi un enter entre 0 i 9 al seu dígit corresponent. Per exemple, per a l'enter 3, cal donar el caràcter '`3`'.

29. Dissenyeu un algorisme que, donats dos intervals reals $[a_1, b_1]$ i $[a_2, b_2]$ digui si el primer es troba dins del segon.

30. Dissenyeu un algorisme que, donats dos intervals reals $[a_1, b_1]$ i $[a_2, b_2]$ calculi l'interval corresponent a la seva intersecció o indiqui que és buida.

31. Dissenyeu un algorisme que, donades dues hores del rellotge $h_1{:}m_1{:}s_1$ i $h_2{:}m_2{:}s_2$, indiqui si una altra hora $h{:}m{:}s$ es troba entre la primera i la segona.

32. Un any de traspàs és un any civil que consta de 366 dies civils. Són anys de traspàs tots els múltiples de 4 que no ho són de 100, o que són de 100 però també de 400.

    Escriviu una expressió que indiqui si una variable entera *any* és de traspàs o no.

33. Dissenyeu un algorisme que llegeixi dos enters i n'escrigui el màxim.

34. Dissenyeu un algorisme que llegeixi dos enters i els escrigui ordenats.

35. Dissenyeu un algorisme que llegeixi tres enters i els escrigui ordenats.

36. Dissenyeu un algorisme que llegeixi quatre enters i els escrigui ordenats.

37. Dissenyeu un algorisme que llegeixi un enter que representa una data en format DDMMAAAA i escrigui el dia, el mes i l'any corresponent.

38. Dissenyeu un algorisme que llegeixi un enter que representa una hora del rellotge en format HHMMSS i escrigui l'hora, els minuts i els segons corresponents.

39. Dissenyeu un algorisme que llegeixi dues dates (dia, mes i any de cadascuna) i digui si la primera és anterior en el temps a la segona.

40. Dissenyeu un algorisme que digui si un enter positiu és primer o no.

41. Dissenyeu un algorisme que escrigui el producte de factors primers d'un número natural.

42. Dissenyeu un algorisme que escrigui les taules de multiplicar del 0 al 9.

43. Dissenyeu un algorisme que intercanviï el valor de dues variables tot usant una variable auxiliar.

44. Dissenyeu un algorisme que intercanviï el valor de dues variables enteres sense usar cap variable auxiliar.

45. Què penseu de l'algorisme següent?

```
int x = ..., i = 10;
while (i > 0) {
    if (x == x) i = i + 1;  else i = i - 1;
}
```

46. Què penseu de l'algorisme següent?

```
int i = ...;
while (i != 0) i = i - 2;
```

47. Simplifiqueu l'algorisme següent:

```
int x = ..., y = ...;
bool iguals;
if (x == y) iguals = true;  else iguals = false;
```

48. Un mag diu: "Pensa't un número. Multiplica'l per dos. Suma-li 34. Divideix-lo per dos. Treu-li el número que havies pensat. Te'n queden 17!"

    Què en penseu?

1.1 La congruència de Zeller per a calendaris

La congruència de Zeller és un càlcul que permet obtenir el dia de la setmana per a una data qualsevol del calendari gregorià. Donada una data determinada pel triplet $\langle d, m, a\rangle$, on d és el dia del mes, m és el mes de l'any i a és l'any,

1. Se li resten dues unitats al mes m i si dóna zero o menys se li suma 12 al mes i se li resta una unitat a l'any. El nou mes obtingut l'anomenem m' i el nou any a'.

2. Es calcula la centúria c (els dos primers dígits de l'any) a partir de l'any a'.

3. Es calcula l'any dins la centúria y (els dos darrers dígits de l'any) a partir de l'any a'.

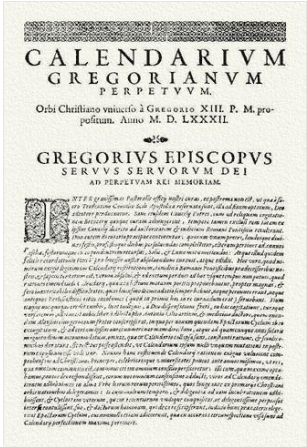
4. Es calcula
$$f = \lfloor 2.6m' - 0.2\rfloor + d + y + \lfloor y/4\rfloor + \lfloor c/4\rfloor - 2c,$$
on $\lfloor x\rfloor$ és la part entera per defecte de x.

5. f mòdul 7 representa el dia de la setmana segons la correspondència següent: 0 = diumenge, 1 = dilluns, ...

Utilitzeu la congruència de Zeller per determinar en quin dia de la setmana vau néixer. Funciona?

Dissenyeu un algorisme que, tot utilitzant la congruència de Zeller, llegeixi una data (dia, mes i any) i escrigui en quin dia de la setmana cau.

1.2 El valor clau per a calendaris


CALENDARIVM
GREGORIANVM
PERPETVVM.
Orbi Christiano vniuerso à Gregorio XIII. P. M. propositum. Anno M. D. LXXXII.
GREGORIVS EPISCOPVS
SERVVS SERVORVM DEI
AD PERPETVAM REI MEMORIAM

Un any de traspàs és un any civil que consta de 366 dies civils. Després de la reforma gregoriana són anys de traspàs tots els anys múltiples de quatre que no acabin en dos zeros i també tots aquells que, acabats en dos zeros, tinguin el nombre que quedaria en treure els dos zeros finals divisible per quatre.

Així, 1700, 1800, 1900, tot i ésser múltiples de 4, no foren de traspàs; en canvi, l'any 2000 ho va ser.

El mètode del *valor clau* és un càlcul que permet obtenir el dia de la setmana per a una data qualsevol en el calendari gregorià. Donat un dia determinat pel triplet $\langle d, m, a\rangle$, on d és el dia del mes, m és el mes de l'any i a és l'any,

1. Es calcula la centúria c (els dos primers dígits de l'any) a partir de l'any a.
2. Es calcula l'any dins la centúria y (els dos darrers dígits de l'any) a partir de l'any a.
3. S'agafa el residu r de dividir c entre 4.
4. Es calcula
$$v_1 = \left\lfloor \frac{y}{4} \right\rfloor + d + 11 - 2r + y$$
on $\lfloor x \rfloor$ és la part entera per defecte de x.
5. Es calcula
$$v_2 = v_1 + clau(m)$$
on $clau(m)$ és una funció que pren els valors següents:

m	1	2	3	4	5	6	7	8	9	10	11	12
$clau(m)$	1	4	4	0	2	5	0	3	6	1	4	6

6. Si el mes m correspon a gener o febrer i l'any a és de traspàs, es resta 1 a v_2.
7. Es calcula s, que val el resultat d'agafar el residu de dividir v_2 entre 7 i afegir-li 1.
8. El valor de s indica el dia de la setmana segons la correspondència següent: 1 = dilluns, 2 = dimarts, ...

Utilitzeu el mètode del valor clau per determinar en quin dia de la setmana vau néixer. Funciona?

Dissenyeu un algorisme que, tot utilitzant el mètode del valor clau, llegeixi una data (dia, mes i any) i escrigui en quin dia de la setmana cau.

2

Seqüències

Exercicis

1. Dissenyeu un algorisme que llegeixi una seqüència d'enters acabada per un zero i digui quants enters la componen.
2. Dissenyeu un algorisme que llegeixi una seqüència d'enters acabada per un zero i compti quants cops apareix el número 123.
3. Dissenyeu un algorisme que llegeixi una seqüència d'enters acabada per un zero i compti quants números positius hi apareixen.
4. Dissenyeu un algorisme que llegeixi una seqüència d'enters acabada per un zero i digui si hi ha més positius que negatius.
5. Dissenyeu un algorisme que llegeixi una seqüència d'enters i digui quants enters la componen.
6. Dissenyeu un algorisme que llegeixi una seqüència d'enters i compti quants cops apareix el número 123.
7. Dissenyeu un algorisme que llegeixi una seqüència d'enters i compti quants números positius hi apareixen.
8. Dissenyeu un algorisme que llegeixi una seqüència d'enters i digui si hi ha més positius que negatius.
9. Dissenyeu un algorisme que llegeixi una frase i digui quants caràcters la componen.
10. Dissenyeu un algorisme que llegeixi una frase i digui quants cops hi apareix la lletra Q.
11. Dissenyeu un algorisme que llegeixi una frase i digui quantes vocals hi apareixen.
12. Dissenyeu un algorisme que llegeixi una frase i digui si hi ha més vocals que consonants.

Part II

Problemes resolts

1

Introducció

1.1 La congruència de Zeller per a calendaris

La resolució d'aquest problema involucra tres etapes:

1. Llegir les dades d'entrada (una data, composta d'un dia d, un mes m i un any a).
2. Realitzar els cinc passos descrits a l'enunciat, tot utilitzant variables auxiliars.
3. Escriure el resultat.

Aquesta és la implementació completa:

```
#include <iostream>
using namespace std;

int main () {
    // Llegir les dades d'entrada
    int d, m, a;  cin >> d >> m >> a;

    // Calcular quin dia de la setmana és amb la congruència de Zeller
    int a2, m2;
    if (m-2 <= 0) {
        m2 = m - 2 + 12;  a2 = a - 1;
    } else {
        m2 = m - 2;       a2 = a;
    }
    int c = a2 / 100;
    int y = a2 % 100;
    int f = (26*m2-2) / 10 + d + y + (y/4) + (c/4) - 2*c;
    int s = f % 7;
    if (s < 0) s += 7;         // Compte!

    // Escriure el dia de la setmana corresponent
         if (s == 0) cout << "Diumenge"  << endl;
```

```
    else if (s == 1) cout << "Dilluns"   << endl;
    else if (s == 2) cout << "Dimarts"   << endl;
    else if (s == 3) cout << "Dimecres"  << endl;
    else if (s == 4) cout << "Dijous"    << endl;
    else if (s == 5) cout << "Divendres" << endl;
    else             cout << "Dissabte"  << endl;
}
```

Observeu que no cal controlar que `s` sigui 6 per escriure `Dissabte`. Fixeu-vos també que el mòdul de C++ no va bé amb els negatius i que cal tenir-ho en compte.

1.2 El valor clau per a calendaris

La solució d'aquest problema segueix la mateixa idea que l'anterior:

```
#include <iostream>
using namespace std;

int main () {
    // Llegir la data
    int d, m, a;  cin >> d >> m >> a;

    // Calcular el dia de la setmana amb el valor clau
    int c = a / 100;
    int y = a % 100;
    int r = c % 4;
    int v1 = y/4 + d + 11 - 2*r + y;
    int v2;
         if (m==1 or m==10)         v2 = v1 + 1;
    else if (m==2 or m==3 or m==11) v2 = v1 + 4;
    else if (m==5)                  v2 = v1 + 2;
    else if (m==6)                  v2 = v1 + 5;
    else if (m==8)                  v2 = v1 + 3;
    else if (m==9 or m==12)         v2 = v2 + 6;
    else                            v2 = v1 + 0;

    bool traspas = a%400==0 or a%4==0 and a%100!=0;
    if (m <= 2 and traspas) v2 = v2 - 1;
    int s = v2%7 + 1;

    // Escriure el dia corresponent
         if (s == 7) cout << "Diumenge"  << endl;
    else if (s == 1) cout << "Dilluns"   << endl;
    else if (s == 2) cout << "Dimarts"   << endl;
    else if (s == 3) cout << "Dimecres"  << endl;
    else if (s == 4) cout << "Dijous"    << endl;
    else if (s == 5) cout << "Divendres" << endl;
    else             cout << "Dissabte"  << endl;
}
```

2

Seqüències

2.1 Misteri

a) El programa escriu 9.

b) Aquest algorisme escriu el màxim d'una seqüència no buida d'enters.

2.2 Vaques boges

Primer observem que darrera el problema a resoldre hi ha una seqüència d'enters acabada amb el sentinella 0 (codi fictici). Aquesta seqüència es defineix com a

$$codi_1, codi_2, \ldots, codi_k, \ldots, 0$$

i la manera d'obtenir els seus elements és:

- ⟨*primer element*⟩: Llegir primer codi (**llegir**($codi_1$)).
- ⟨*següent element*⟩: Llegir següent codi (**llegir**($codi_{k+1}$)).
- ⟨*darrer element*⟩: Mirar si el codi actual és 0 ($codi_k = 0$).

Observem ara que per a la solució del problema cal aplicar l'esquema de cerca sobre la seqüència, tot buscant un element de la seqüència que és múltiple de 13 i de 15:

```
int main () {
    bool trobat = false;
    int codi;  cin >> codi;                      // Obtenir primer element
    while (codi != 0 and not trobat) {           // Mirar si és darrer element
        if (codi%13 == 0 and codi%15 == 0) {     // Mirar condició de cerca
            trobat = true;
        } else {
            cin >> codi;                         // Obtenir següent element
        }
```

```
    }
    if (trobat) cout << "Alerta: Hi ha vaques boges." << endl;
    else  cout << "Tranquils: no hi ha vaques boges." << endl;
}
```

2.3 El control de qualitat

Per resoldre aquest problema s'ha d'aplicar l'esquema de cerca a la seqüència de pesos de les peces $pes_1, pes_2, \ldots, pes_k, \ldots, -1$. Es cerca una peça que no passi el control de qualitat. La forma d'obtenir els elements de la seqüència és:

- ⟨*primer element*⟩: **llegir**(pes_1)
- ⟨*següent element*⟩: **llegir**(pes_{k+1})
- ⟨*detecció final seqüència:*⟩: quan no s'ha pogut llegir.

Ara dissenyem l'algorisme tenint en compte que la propietat de cerca és que el pes actual no estigui entre 100 i 150 grams, que són respectivament el *pesMinim* i el *pesMaxim* que pot tenir una peça:

```
const int PES_MINIM = 100, PES_MAXIM = 150;

int main () {
    bool trobat = false;
    int pes;
    while (cin >> pes and not trobat) {
        if (pes < PES_MINIM or pes > PES_MAXIM) trobat = true;
    }
    if (trobat) cout << "NO" << endl;  else cout << "SI" << endl;
}
```

Noteu que en cas de trobar una peça que no passa el control de qualitat la resposta ha de ser "`NO`" i que en cas de no trobar cap la resposta ha de ser "`SI`".

2.4 Infraccions

L'algorisme que hem de dissenyar s'obté aplicant l'esquema de recorregut sobre la seqüència dels elements $\langle he, hs\rangle$ que es llegeix tot observant que l'element sentinella és $\langle -1, -1\rangle$. La manera d'obtenir els elements de la seqüència és:

- ⟨*primer element*⟩: **llegir**(he_1, hs_1)
- ⟨*següent element*⟩: **llegir**(he_{k+1}, hs_{k+1})
- ⟨*darrer element*⟩: $he_k = -1$ **i** $hs_k = -1$

Quant al tractament de l'element de la seqüència, hem de fer:

- Comptar un vehicle més (a fi de calcular el temps d'estada mitjana).
- Sumar el temps d'estada del vehicle, o sigui la diferència entre l'hora de sortida i l'hora d'entrada, al temps total d'estada (a fi de calcular el temps d'estada mitjana). Noteu que podem restar les hores de sortida i entrada directament ja que vénen donades en minuts (nombres enters).