

Disseny de Jocs de Proves

PRO1

Josep Carmona, Lluís Padró

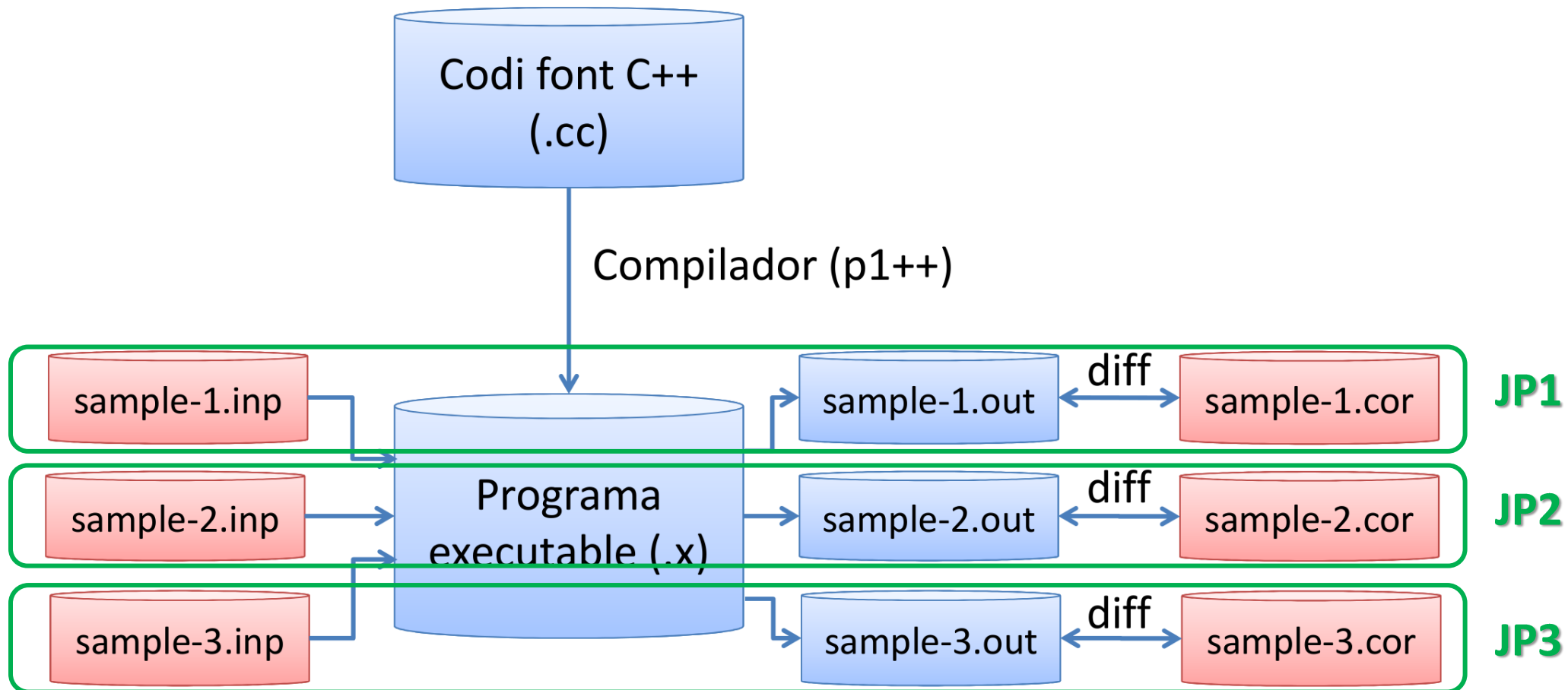


UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Index

- **Breu descripció**
- Pas 1: Fer servir els JPs que ens facilita el jutge
- Pas 2: Dissenyar JPs propis
- Exercici

Els Jocs de Proves (JP)



Els Jocs de Proves (JP)

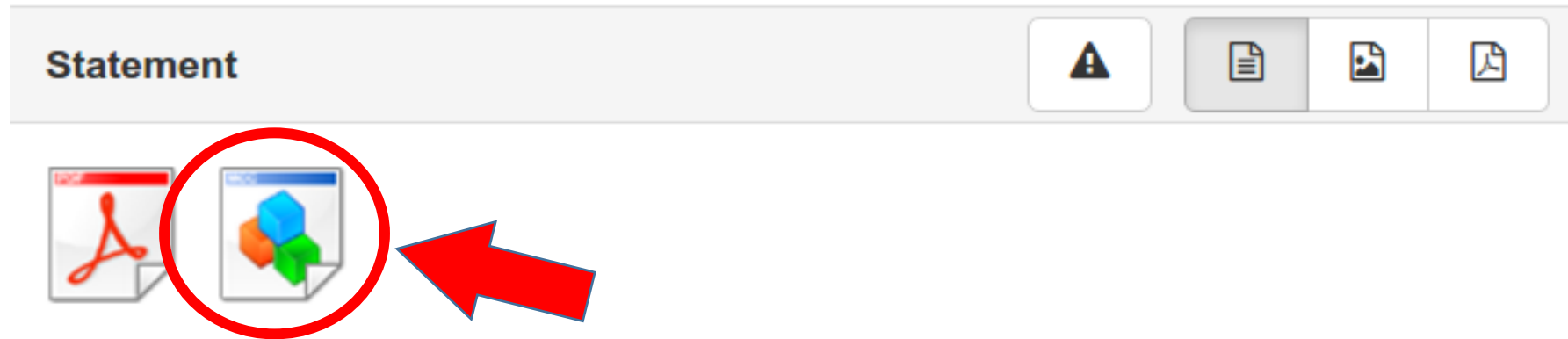
- Permeten tenir una certa confiança respecte la correctessa dels programes (però no un 100% de confiança!).
- Eviten enviaments al jutge.org. Recordeu que la normativa actual d'avaluació de **controls** de l'assignatura **penalitz**a a partir del tercer enviament.
- Acostumar-se a dissenyar JPs és una molt bona pràctica, que us servirà en el futur (dintre i fora de la carrera), ja que no hi ha jutge a la vida real.

Index

- Breu descripció
- **Pas 1: Fer servir els JPs que ens facilita el jutge**
- Pas 2: Dissenyar JPs propis
- Exercici

Fer servir els JPs del jutge

1. Baixar els fitxers del problema a la teva zona personal:



Write a program to compute powers.

Input

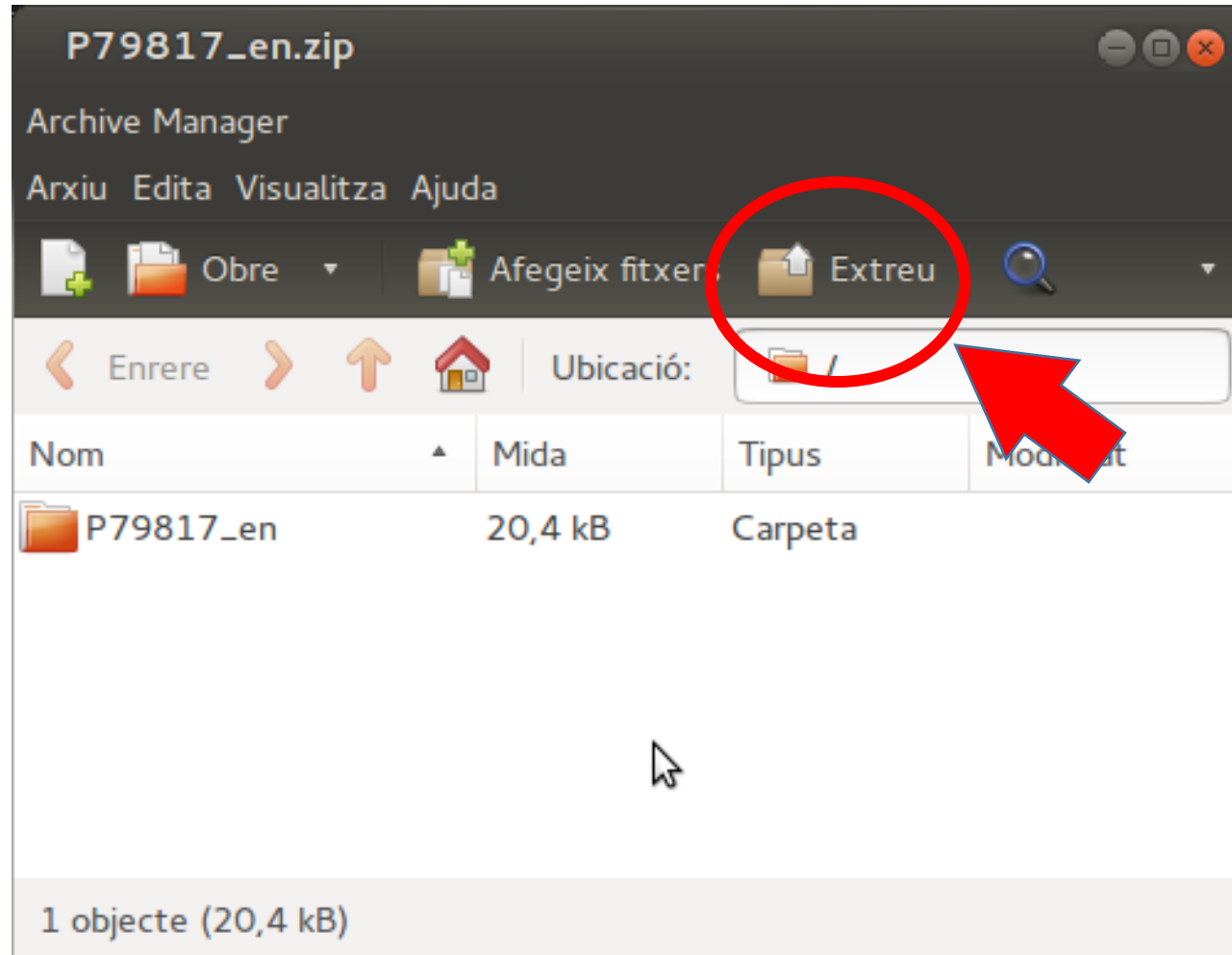
Input consists of several pairs of integer numbers a and b . Assume $b \geq 0$.

Output

For every pair a, b , print a^b . Suppose, as usual, that $0^0 = 1$.

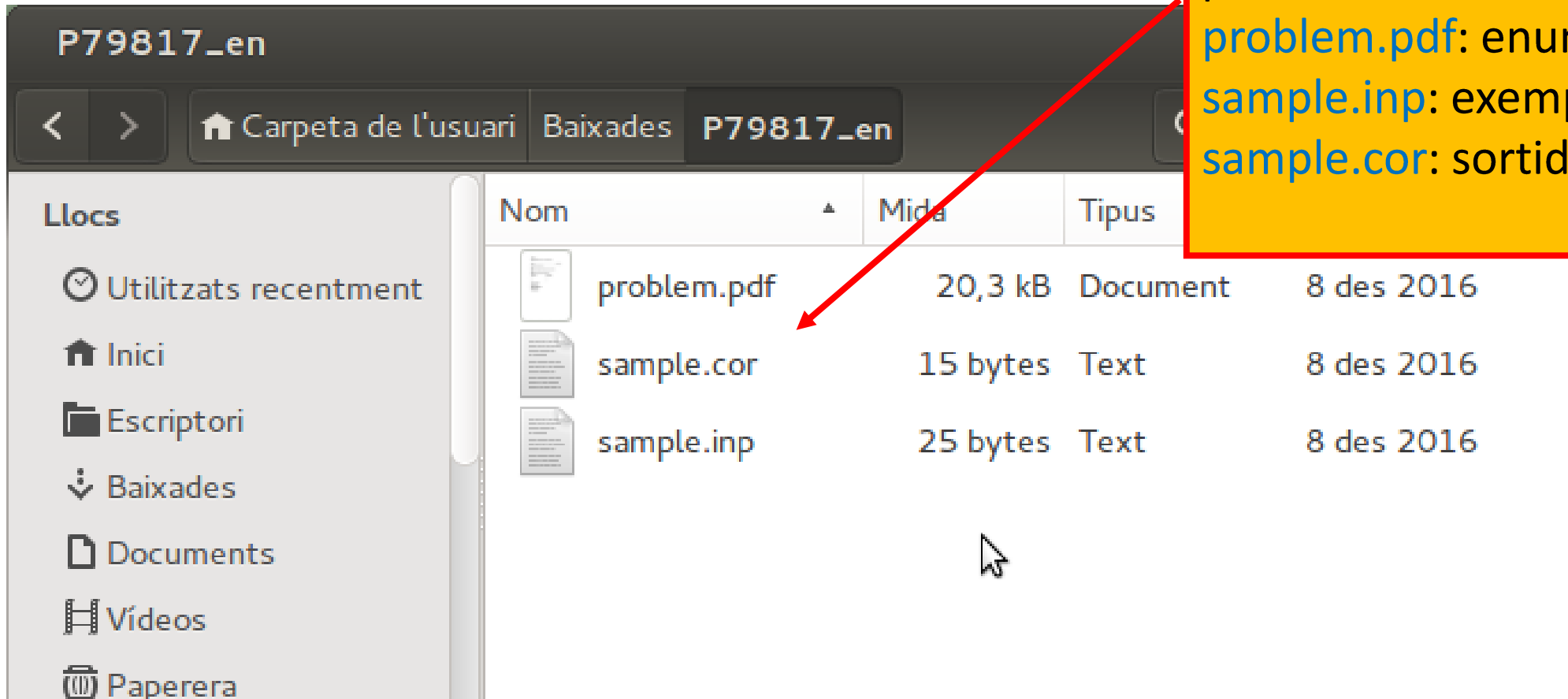
Fer servir els JPs del jutge

2. Descomprimir el fitxers del problema:



Fer servir els JPs del jutge

2. Descomprimir el fitxers del problema:



Un cop descomprimit, tenim una carpeta amb la informació del problema:

[problem.pdf](#): enunciat

[sample.inp](#): exemple d'entrada

[sample.cor](#): sortida esperada

Fer servir els JPs del jutge

3. Executar el teu programa guardant la sortida en un arxiu.
4. Comparar la sortida del teu programa amb la sortida esperada.

```
padro@sukania: ~/Baixades/P79817_en
Fitxer Edita Visualitza Cerca Terminal Ajuda
~/Baixades/P79817_en$ ./P79817.x < sample.inp > sample.out
~/Baixades/P79817_en$ kompare sample.out sample.cor
```

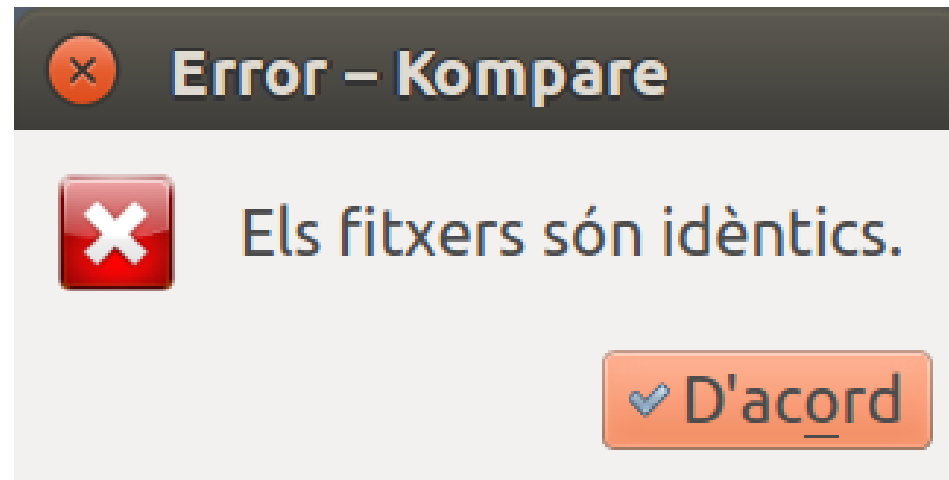
Comparo la sortida del programa (`sample.out`) amb la sortida esperada `sample.cor` mitjançant el programa **kompare**

Executo el programa `P79817.x` amb entrada `sample.inp` i deixant la sortida en `sample.out`. Si `sample.out` existeix es sobreescrirà.

Fer servir els JPs del jutge

5. Sortida de **kompare**.

A) programes generen mateixa sortida



Fer servir els JPs del jutge

5. Sortida de **kompare**. B) Existeixen discrepàncies

file:///home/padro/Baixades/P79817_en/sample.out -- file:///home/padro/Baixades/P79817_en/sample.cor

File Difference Settings Help

Compare Files Save Save All Previous File Next File Previous Difference

Navigation

Source Folder	Destination Folder	Source File	Destination File	Source L	Destination	Difference
/home/padro...	/home/padro/B...	sample.out	sample.cor	1	1	Changed 1 line
				5	5	

sample.out sample.cor

1 19	19
2 256	2 256
3 1	3 1
4 0	4 0
5 81	5 1
6	6 -8

Comparing file file:///home/padro/B...padro/Baixades/P79817_en/sample.cor 1 of 2 differences, 0 applied 1 of 1 file

Diferències:

Linia 1: El programa escriu 19, però hauria de ser un 9

Linia 5: El programa escriu 81, però hauria de ser un 1

Linia 6: Falta un -8 que el programa no escriu

Index

- Breu descripció
- Pas 1: Fer servir els JPs que ens facilita el jutge
- **Pas 2: Dissenyar JPs propis**
- Exercici

Dissenyar els JPs propis

- Com a regla general, un bon JP ha de cobrir tot el codi.
- Anem a veure-ho amb el codi per decidir si un nombre és primer.

Prime number

```
// Input:  read a natural number N>0
// Output: write "is prime" or "is not prime" depending on
//         the primality of the number
```

```
int main() {
    int N;
    cin >> N;

    int divisor = 2;
    bool is_prime = (N != 1);
    // 1 is not prime, 2 is prime, the rest enter the loop (assume prime)

    // is_prime is true while a divisor is not found
    // and becomes false as soon as the first divisor is found
    while (divisor < N) {
        if (N%divisor == 0) is_prime = false;
        divisor = divisor + 1;
    }

    if (is_prime) cout << "is prime" << endl;
    else cout << "is not prime" << endl;
}
```

EXECUTION 1: N is not a prime number

Prime number

```
// Input:  read a natural number N>0
// Output: write "is prime" or "is not prime" depending on
//         the primality of the number
```

```
int main() {
    int N;
    cin >> N;

    int divisor = 2;
    bool is_prime = (N != 1);
    // 1 is not prime, 2 is prime, the rest enter the loop (assume prime)

    // is_prime is true while a divisor is not found
    // and becomes false as soon as the first divisor is found
    while (divisor < N) {
        if (N%divisor == 0) is_prime = false;
        divisor = divisor + 1;
    }

    if (is_prime) cout << "is prime" << endl;
    else cout << "is not prime" << endl;
}
```

EXECUTION 2: N is a prime number

Dissenyar els JPs propis

- Cal llegir bé l'enunciat per determinar aquells casos extrems que compleixen l'enunciat però que potser requereixen un tractament especial.
- Anem a veure uns exemples d'aquestes situacions.

Dissenyar els JPs propis

Statement



Write a program that, given a number n , prints a “triangle of size n ”.

Input

Input consists of a natural number n .

Que ha d'imprimir el programa si $n=0$?

Output

Print n lines, in such a way that the i -th line contains i asterisks.

Dissenyar els JPs propis

Statement



Write a program that reads two numbers a and b , and prints all numbers between a and b .

Input

Input consists of two natural numbers a and b .

Output

Print a line with $a, a+1, \dots, b-1, b$. Separate the numbers with commas.

Que ha d'imprimir el programa si:

- 1) $a = b$
- 2) $a > b$

Dissenyar els JPs propis

Statement



Write a program that reads a list of words and prints the number of times that the word “he`l`lo” appears in it.

Input

There input is a sequence of strings.

Output

Print the number of occurrences of the word “he`l`lo”.

Observation

Take into account that “ho`l`a” is different from “Ho`l`a”.

Que ha d'imprimir el programa si:

- 1) La seqüència d'entrada és buida
- 2) La seqüència d'entrada no conté la paraula “hello”
- 3) La seqüència d'entrada conté “Hello”

Index

- Breu descripció
- Pas 1: Fer servir els JPs que ens facilita el jutge
- Pas 2: Dissenyar JPs propis
- **Exercici**

Exercici

- Feu l'exercici X50286 ("*How many hello?*"), aplicant el que heu après en aquest tutorial.