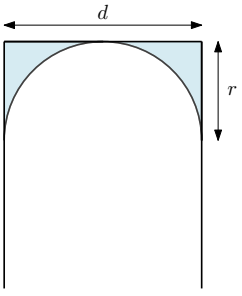
Optimus Aquæductus

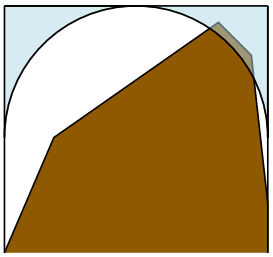
El gobernador de la provincia tarraconense ens ha demanat si li podem dissenyar aqueductes amb el mínim cost. Per això, ens donarà les alçades de les elevacions que hem de superar entre dos punts, i a quina alçada ha d'estar l'aqueducte per a que pugui portar l'aigua a la ciutat.

Els aqueductes romans es fan amb arc de mig punt (semicircumferència). Aquest estil clàssic d'aqueducte es recolza en pilars que s'estenen des del terra fins l'aqueducte. Els arcs entre pilars distribueixen el pes de l'aqueducte als pilars adjacents.

Els aqueductes que construim son pilars espaiats a intervals irregulars, i sempre tenen arcs de mig punt, tal com es mostra a la figura 1a. Tanmateix, tot i que un arc pot tocar terra, no es pot estendre per sota terra. Això fa que algunes ubicacions de pilars siguin impossibles (figura 1b).



(a) Els pilars consecutius a distància d estan units per un arc de mig punt de radi $r = d/2$.



(b) Una col·locació de pilar no vàlida (els arcs no es poden estendre per sota terra).

Donat un perfil terrestre i l'alçada desitjada del pont h, normalment hi ha moltes maneres de construir un aqueducte. Modelem el perfil del sòl com una funció lineal a trossos descrita per n punts clau $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, on la coordenada x d'un punt és la posició al llarg del pont, i la coordenada y és l'elevació del sòl sobre el nivell del mar en aquesta posició al llarg del pont. El primer i l'últim pilars s'han de construir al primer i a l'últim punt clau i els pilars intermedis només es poden construir en aquests punts clau. El cost d'un pont és el cost dels seus pilars (que és proporcional a les seves alçades) més el cost dels seus arcs (que és proporcional a la quantitat de material utilitzat). Així doncs, un pont amb k pilars d'alçada $h_1, \ldots, h_k$ que estan separades per distàncies horitzontals $d_1, \ldots, d_{k-1}$ té un cost total de

$$\alpha \sum_{i=1}^{k} h_i + \beta \sum_{i=1}^{k-1} d_i^2$$

per a les constants donades α i β. L'objectiu és construir cada aqueducte al menor cost possible.

Format d'entrada

La primera línia d'entrada conté quatre enters n, h, α i β, on n ($2 \leq n \leq 10^4$) és el nombre de punts que descriuen el perfil del sòl, h ($1 \leq h \leq 10^5$) és l'alçada desitjada de l'aqueducte per sobre el nivell del mar i α, β ($1 \leq \alpha, \beta \leq 10^4$) són els factors de cost, tal com es s'ha descrit anteriorment. A continuació hi ha n línies, on la línia i conté dos enters x_i, y_i ($0 \leq x_1 < x_2 < \cdots < x_n \leq 10^5$ i $0 \leq y_i < h$), descrivint el perfil del sòl.

Format de sortida

Treurem el cost mínim de construir un pont des de la posició horitzontal x_1 fins a x_n a l'alçada h sobre el nivell del mar. Si és impossible construir cap pont d'aquest tipus, la sortida serà `impossible`.

Requeriments

Es demana realitzar:

- Tres programes que llegeixin les dades d'un fitxer de text.
  - Greedy: un mètode voraç que trigui un temps $\mathcal{O}(n)$, on n és el nombre de punts al sòl. La solució no cal que sigui òptima.
  - Backtracking: un mètode segons l'esquema de retrocès amb un temps òptim, donant una soluició òptima.
  - Dynamic Programming: un mètode segons l'esquema de programació dinàmica, donant una solució òptima.

  La codificació haurà de superar el test del `makefile`.
- Un informe que contingui, sobre els algorismes principals: pseudo-codi de l'algorisme (pot ser recursiu o iteratiu), cost teòric i experimental.

Arguments i parametres

L'execuci'o de l'aplicació haurà de seguir la següent sintaxi:

```
$./aqueducte <fitxer entrada>
```

Els arguments opcionals del programa són els següents:

fitxer entrada : Fitxer amb les dades d'entrada.

Si no hi ha arguments opcionals, l'aplicació llegeix de l'entrada estàndard i escriu a la sortida estàndard.

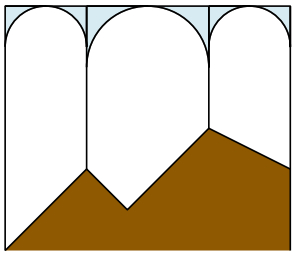


Figura 2: Un pont corresponent a l'entrada 1.

Exemples

Aquests són dos exemples amb una entrada i la sortida corresponent.

Entrada 1	Sortida 1

`5 60 18 2`
`0 0`
`20 20`
`30 10`
`50 30`
`70 20`
	`6460`

Entrada 2	Sortida 2

`4 10 1 1`
`0 0`
`1 9`
`9 9`
`10 0`
	`impossible`

A la figura 2 teniu la representació corresponent al primer exemple .

Avaluació

En l'informe de la pràctica, els algorismes principals, les taules i les gràfiques han d'estar comentats.

Si hi ha codi que s'ha de compilar, els programes han de compilar sense errors ni warnings i passar els tests amb `make test`, altrament no s'avaluarà.

La baremació de la pràctica (sobre 10) serà la següent:

Anàlisi de costos 3 punts

Disseny 3 punts, Es tindrà en compte la dificultat, i el cost de l'algorisme;

Implementació 4 punts

- 1 punt, iteratiu en python (opcionalment en C/C++)
- 1 punt, recursiu en python (opcionalment en Haskell). La recursivitat només s'ha d'implementar en l'algorisme principal.
- 1 punt, bones pràctiques de programació (`pylint`)
- 1 punt, utilització dels recursos del llenguatge de programació.

Enviament

L'assignació és per parelles, i representa un 30% de la nota final. Presenteu la pràctica al Campus Virtual de la UdL amb dos fitxers: un pdf per a l'informe, i un arxiu comprimit, `zip` (no s'admeten altres formats de compressió), per al codi font.

A l'arxiu comprimit hi haurà un `README.md` amb una breu explicació del codi (pot ser part del contingut del document), i ha de tenir l'enllaç al vostre repositori a `github.com`. L'informe, de fet, pot ser el `README.md` passat a pdf.

Els programes que s'hagin de compilar ho han de fer amb:

```
$ make
```

Els programes es testejen fent:

```
$ make test
```

La pràctica podria ser verificada individualment el primer dia de laboratori després de l'entrega.