Cognoms, Nom (Amb lletres majúscules)	D.N.I.

Titulació: Grau en Enginyeria Informàtica

Curs: Q2 2022–2023 (1r Parcial)

Assignatura: Programació 2 (PRO2)

Data: 24 d'abril de 2023

Duració: 2h

1. (5 punts) Suposeu que ens donen l'expansió decimal dig d'un número amb moltes xifres, p.e., els primers 100000 dígits de  $\pi$ , en un vector<int>, cada component del vector conté un dígit; també ens donen un enter positiu  $n \geq 0$  i el número k de xifres decimals que té n. Volem una funció quantes\_vegades\_dintre que ens diu quantes vegades apareix n dintre de l'expansió decimal donada dig. Suposem per exemple que ens donen

$$dig = [3, 1, 4, 1, 5, 9, 2, 6, 5, 4, 1, 6, 4, 2, 2, 2, 2]$$

Alguns resultats que ens retonaria la funció serien:

- quantes\_vegades\_dintre(dig, 592, 3) = 1, ja que 592 apareix una vegada, a la posició 4, concretament dig[4..6] = 592
- quantes\_vegades\_dintre(dig, 41, 2) = 2, doncs 41 apareix dos cops, a les posicions 2 i 9 de dig,
- quantes\_vegades\_dintre(dig, 598, 3) = 0, ja que el número 598 no apareix enlloc del vector dig,
- quantes\_vegades\_dintre(dig, 22, 2) = 3, 22 apareix 3 cops, començant a les posicions 13, 14 i 15 del vector dig.

Implementa iterativament la funció quantes\_vegades\_dintre amb aquesta capçalera

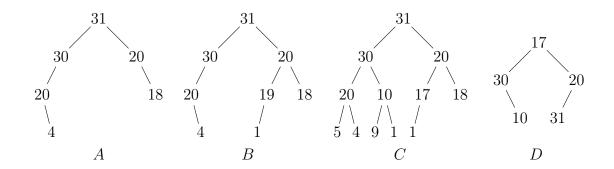
```
// Pre: n \geq 0 \land 0 < k = \text{número de xifres d'} n \leq dig.\text{size}() // \land \forall i : 0 \leq i < dig.\text{size}() : 0 \leq dig[i] \leq 9 int quantes_vegades_dintre(const vector<int>& dig, int n, int k); // Post: retorna el nombre de vegades que les xifres d'n apareixen // com a subseqüència de dig
```

La teva solució ha de ser eficient. No has d'extreure els dígits d'n i posar-los en un vector en cap cas. Sí que pots convertir subseqüències del vector dig en el número decimal al que representen. Pots fer ús de qualsevol operació aritmètica sobre els enters i la funció pot(x,y) per a calcular  $x^y$ , però convé no fer més crides de les estrictament necessàries, doncs el cost en temps de la funció pot és prou elevat.

Escriu a) l'invariant i b) la funció de fita del bucle principal del teu algorisme, i argumenta la seva correcció: c) justifica la inicialització, d) justifica la condició de sortida, e) justifica el cos del bucle i f) justifica que el bucle acaba.

2. (5 punts) Diem que un arbre binari a (BinTree<T>) està contingut a un altre arbre binari b si i només si tots els nodes d'a apareixen a l'arbre b amb idèntics valors i als llocs corresponents, començant per l'arrel als dos arbres.

Donats dos BinTree<int>s a i b, escriu una funció booleana esta\_contingut(a,b) que retorna cert si i només si a està contingut en b. Per exemple, en la figura més avall, l'arbre A està contingut a l'arbre B i també a l'arbre C; però, ni B conté a C, ni C conté a B. D'altra banda, A no conté ni a B ni a C. L'arbre D no conté a cap dels altres ni està contingut a cap dels altres.



Escriu la capçalera i especificació de la funció esta\_contingut i implementa-la. Demostra l'acabament i correctesa de la teva implementació donant: 1) una funció de mida, 2) les hipòtesis d'inducció, 3) justificació del seu acabament, 4) justificació del(s) cas(os) base i 5) justificació del(s) cas(os) recursiu(s).

Es valorarà que la funció implementada explori la part dels arbres a i b estrictament necessària per a determinar si a està contingut a b o no.