Control Grups EF Pràctica 2: Estructures Lineals

Nom i Cognoms: _____ **Firma:** _____

Abans de començar:

- **CAMPUS VIRTUAL**: descarrega't el codi de la teva pràctica 2 del Campus Virtual i descomprimeix-lo.
- En aquest control s'ha d'utilitzar **NetBeans 8.2, C++ versió 11**.
- S'ha de lliurar al campus virtual **un únic fitxer zip** amb el vostre nom i cognoms. Per exemple, LisaSimpson_control_P2.zip que contindrà el codi dels dos exercicis. Cada exercici és un projecte NetBeans.

(2.5 PUNTS) EXERCICI 1: En el TAD ArrayDeque, afegireu i implementareu una nova funció anomenada:

```
void enqueueWithPriority(const int value, int priority);
```

Aquesta funció, donada una prioritat, inseriria el nou element tantes posicions endavant com indiqui el valor de la prioritat. En el cas que la priority sigui un valor més gran que el nombre d'elements o sigui un valor negatiu, s'ha de llençar una excepció.

A continuació mostrem un exemple, si la cua té els valors [1, 2, 3], el resultat d'aplicar posteriorment `enqueueWithPriority(4,2)`, deixarà la cua amb els valors [1, 4, 2, 3]. Noteu que volem inserir l'element 4 amb una prioritat d'avançar 2 posicions, respecte la posició que li pertocaria inicialment a la cua. Com sabem, si féssim servir el mètode enqueueBack que ja tenim implementat, aquest element s'inseriria al final (ocuparia la posició 3 dins l'array); en canvi, aquest cop utilitzarem el mètode `enqueueWithPriority(4, 2)`, o el que és el mateix, inserirem l'element 4 amb prioritat 2. El resultat d'aquesta operació es tradueix en inserir l'element dues posicions endavant del que normalment ho hauria de fer. Així doncs, l'element en compte d'ésser inserit a la posició 3 de l'array serà incorporat a la posició 1.

- Definiu el nou mètode en el TAD definit en el projecte de **l'exercici 1** de la pràctica. Es valorarà positivament que feu la millor implementació possible i tingueu en compte el cost computacional.
- Afegiu a l'inici del main el següent tros de codi, tot mantenint després el codi que vàreu fer per ArrayDeque:

```
ArrayDeque q(5);
int elements[3] = {1, 2, 3};
for (int i = 0; i < 3; i++) q.enqueueBack(elements[i]);
q.print();
q.enqueueWithPriority(4, 2); q.print();
q.enqueueWithPriority(5, 4); q.print();

try {
    q.enqueueWithPriority(6, 1);
} catch (<tipus> e) { // agafeu aquí la vostra excepció, segons el tipus que poseu
   // imprimiu el missatge de l'excepció
 }
q.dequeueFront();
q.enqueueWithPriority(7, 0);
```

- La sortida del main anterior és la següent:

SORTIDA PER PANTALLA

```
[1,2,3]
enqueueWP  element 4 priority 2 size deque 3
[1,4,2,3]
enqueueWP  element 5 priority 4 size deque 4
[5,1,4,2,3]
DeQueue is full!
enqueueWP  element 7 priority 0 size deque 4
[1,4,2,3,7]
```

(3 PUNTS) EXERCICI 2: : En el TAD LinkedDeque, afegireu i implementareu una nova funció anomenada:

```
void enqueueDecimalNumber();
```

Aquesta funció calcula el nombre decimal que formen els elements de la cua i afegeix un nou node al final de la cua amb aquest nombre decimal. Llença una excepció si algun element no està entre 1 i 9.

- Definiu i implementeu la funció en el projecte NetBeans de **l'exercici 2** de la pràctica
- Afegiu a l'inici del main el següent tros de codi, tot mantenint després el codi que vàreu fer per la LinkedDeque

```
LinkedDeque q;
 int elements[5] = {1,2,3,2,4};

for (int i = 0; i < 5; i++) q.enqueueBack(elements[i]);
q.print();
q.enqueueDecimalNumber();  q.print();
```

- La sortida del main anterior és la següent:

```
[1,2,3,2,4]
[1,2,3,2,4,12324]
```

IMPORTANT

- **No podeu usar les funcions del TAD LinkedDeque per implementar aquesta funció**
- **Els recorreguts han de ser amb punters, no s'han de fer amb size**
- **La funció** `pow (base, exponent)` **us permetrà calcular base elevat a exponent. Heu d'incloure la llibreria** `#include <math.h>`