SESSIÓ 1 | Repàs PRO1

Llibreries i tal:

```
using namespace std;
#include <iostream>
#include <vector>
#include <string>
#include <cmath>
#include <algorithm>
```

```
pro2/sessio1> p2++ -c PRO2.cc
pro2/sessio1> g++ -o PRO2.exe PRO2.o
pro2/sessio1> ./PRO2.exe
```

Crea PRO2.o
Crea PRO2.exe
Executa PRO2.exe

Ho podem executar tot en una sola línea així:

`p2++ -c PRO2.cc && g++ -o PRO2.exe PRO2.o && ./PRO2.exe`

Si volem executar `PRO2.exe` amb les dades del fitxer `PRO2.dat`:

```
./PRO2.exe < PRO2.dat
```

Si a més volem que els resultats s'escriguin en un altre fitxer PRO2.sal, executarem:

```
./PRO2.exe < PRO2.dat > PRO2.sal
```

Si volem comprimir un `fitxer.tar`:

```
tar cf nom.tar fitxer1.cc fitxer2.hh fitxer3.exe…
```

Si volem extreure un `fitxer.tar`:

```
tar xf nom.tar
```

SESSIÓ 2 | Piles i Cues

Piles:

```
#include <stack>
stack<tipus_dada> nom_pila;
```

`pila.push(variable);`	Afegeix variable dalt la pila
`pila.pop();`	Treu el que hi hagi dalt la pila
`pila.empty();`	Retorna true si està buida
`pila.top();`	Retorna la variable que hi hagi **dalt** la **pila**
`pila.size();`	Retorna la mida de la pila

Cues:

```
#include <queue>
queue<tipus_dada> nom_cua;
```

`cua.push(variable);`	Afegeix variable a la cua
`cua.pop();`	Treu el que hi hagi primer a la cua
`cua.empty();`	Retorna true si està buida
`cua.front();`	Retorna la variable de **l'inici** de la **cua**
`cua.size();`	Retorna la mida de la cua

SESSIÓ 3 | Llistes, Mapes i Sets

Llistes:

```
#include <list>
list<tipus_dada> nom_llista;

list<tipus_dada>::iterator it;
```

`llista.begin()`	Retorna un iterador apuntant a **l'inici**
`llista.end()`	Retorna un iterador apuntant al **final**
`advance(it, numero);`	
`llista.insert(it, variable);`	Afegeix variable a la posició apuntada per **it**
`llista.erase(it);`	Treu l'element de la posició **it,** retorna un iterador apuntant a la **següent posició**
`llista.front();`	Retorna la variable que hi hagi a **l'inici**
`llista.back();`	Retorna la variable que hi hagi al **final**
`llista.push_front(variable);`	Afegeix variable a **l'inici**
`llista.push_back(variable);`	Afegeix variable al **final**
`llista.pop_front(variable);`	Treu l'element que hi hagi a **l'inici**
`llista.pop_back(variable);`	Treu l'element que hi hagi al **final**
`llista.remove(variable);`	Treu tots els elements amb el **valor** variable
`llista.clear();`	Borra tota la llista
`llista.size();`	Retorna la mida de la llista
`llista.sort();`	**Ordena** la llista

Mapes:

```
#include <map>
map<clau_tipus_dada, valor_tipus_dada> nom_mapa;
map<clau_tipus_dada, valor_tipus_dada>::iterator it;
```

`mapa.begin()`	Retorna un iterador apuntant a **l'inici**
`mapa.end()`	Retorna un iterador apuntant al **final**
`mapa.insert({clau, valor});`	**Afegeix** una parella clau-valor al mapa
`mapa.erase(it);`	Treu l'element de la posició **it,** retorna un iterador apuntant a la **següent posició**
`mapa.at(clau);`	Accedeix al valor associat a la clau donada (Si no existeix fa excepció out_of_range)
`mapa[clau];`	Accedeix al valor associat a la clau donada (Si no existeix la crea)
`mapa.count(clau);`	Retorna el nº d'elements amb clau (0 o 1)
`mapa.find(clau);`	Retorna un iterador apuntant a la clau (si no el troba torna mapa.end())
`mapa.size();`	Retorna la mida del mapa
`mapa.clear();`	Borra tots els elements del mapa

SESSIÓ 4, 5 i 6 | BinTree

BinTree:

```
#include "BinTree.hh"
BinTree<tipus_dada> nom_arbre;
BinTree<tipus_dada> nom_arbre(valor, fillEsquerre, fillDret);
```

`arbre.empty();`	Retorna true si l'arbre és buit
`arbre.left();`	Retorna el subarbre esquerre
`arbre.right();`	Retorna el subarbre dret
`arbre.value();`	Retorna el valor del node actual

SESSIÓ 7 | Modularitat

Fitxers d'Encapçalament (.hh):

- Contenen capçaleres de funcions i definicions de tipus de dades.
- Hi ha dos tipus:
    - Fitxers que creen nous tipus de dades (es criden amb `dada.funció()`).
    - Fitxers que contenen únicament la implementació a funcions.

Fitxers de Codi (.cc):

- Contenen la implementació de funcions i mètodes.

Makefile:

```
all: program.exe                              #Objectiu principal makefile

#Construeix l'executable a partir dels fitxers objectes
program.exe: main.o altres.o                  #Crida a main.o i altres.o
      p2++ -o program.exe main.o altres.o

altres.o: altres.cc altres.hh                 #Compila els objectes
      p2++ -c altres.cc -o altres.o

main.o: main.cc llibreries.hh altres2.hh      #Compila main amb llibreries
      p2++ -c main.cc -o main.o

clean:                                        #Esborra els fitxers temporals
      rm -f *.o                               El cridem amb "make clean"
      rm -f program.exe
```