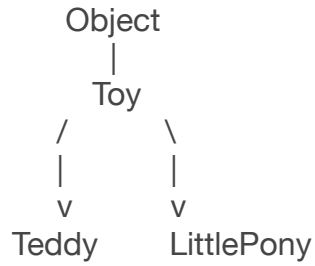
RUSH 2

Makefile

/src/objects/



Class Object : Object.cpp/hpp
- methods:
  - **Object**(std::string const &name)
- attributes **PROTECTED**:
  - _name

Class Toy : hérite de **Object** Toy.cpp/hpp
- methods:
  - **Toy**(std::string const &name)
  - virtual void **isTaken()**
- attributes **PROTECTED**:
  - bool _isTaken

Class Teddy : hérite de **Toy** Teddy.cpp/hpp
- methods:
  - **Teddy**(std::string const &name)
  - void **isTaken() —> affiche** « gra hu »

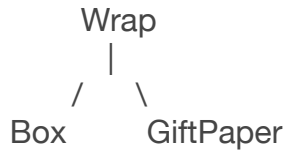
Class LittlePony : hérite de **LittlePony** LittlePony.cpp/hpp
- methods:
  - **Littlepony**(std::string const &name)
  -void **isTaken() —> affiche** « ya man »

Possibilité de créer un **LittlePony** avec un nom (idem pour **Teddy**)
^
|_ *Object **MyUnitTests();*
fonction à part qui permet d'effectuer des tests
en appelant un Teddy(« cuddles ») et un LittlePony(« happy pony »)

Tip: Object **objs = new *Object[2]
objs[0] = new Teddy(« cuddles »)
objs[1] = new LittlePony(« happy pony »)
return objs

##

Il va falloir créer une classe **Elf** qui va accueillir les *Objects* && les *Wraps*



Class Wrap : Wrap.cpp/hpp

- methods:
  - **Wrap()** —> affiche « whistles while working »
  - virtual **wrapMeThat**(Object *) = 0 (virtuelle pure pour Box & GiftPapee)
  - virtual void **openMe**() —> ouvre la *box* (_opened = true)
  - virtual void **closeMe**() —> ferme une *open box* (_opened = false)
  - bool **isOpen**() renvoie _**open**;
  - (pour Elf) virtual Object ***takeMe**()
- attributes (protected):
  - **bool** _opened
  - **Object *_in**

Class Box : hérite de WRAP Box.cpp/hpp

- methods
  - **Box**(name)
  - virtual **closeMe**() —> ferme une *open box* (_opened = false)
  - override **wrapMeThat**(Object *): emballe si _open **true** et **_in NULL**
- attributes

Class GiftPaper : hérite de WRAP GiftPaper.cpp/hpp

- methods
  - Object ***takeMe**() : retourne un Object * et met le _in **NULL**
  - override **wrapMeThat**(Object *): n'a pas besoin d'être _open **true** pour emballer

^
|_Object *MyUnitTests(Object **objs);

Info: objs[0] est **Teddy**
objs[1] est **Box**
objs[2] est **GiftPaper**

Tip: Initialiser des pointeurs
GiftPaper **null**
Box **null**
Toy

NE PAS OUBLIER DE GERER LES ERREURS ET LES AFFICHER DANS LE **ERROR OUTPUT**

Comment emballer un jouet dans une boite et emballer le tout dans du papier cadeau:

- Box->openMe()
- // Box->takeMe() // pour vider la “boite“ et qu’elle soit à nouveau utilisable … histoire de pointeur à NULL
- Box->wrapMeThat(jouet)
- Box->closeMe()
- GiftPaper->takeMe()
- GiftPaper->wrapMeThat(box)
- Retourner le GiftPaper

##

/src/workstation

Class ITable: ITable.cpp/hpp

- methods:
  - ~ITable()
  - virtual void **put**(Object *) = 0 // increment index
  - virtual Object ***take**() = 0 // renvoie les objets présents
  - virtual Object ***take**(int index) = 0 // renvoie l’objet présent à l’index *index*
  - virtual Object ****look** = 0 // Retourne les objets présents (limite = 10 et le dernier est **NULL**)
- attributes (protected):
  - Object **_objs
  - int _objNumber // si ce **nombre > 10**, la **table s’écrase, += 1 au put**
  - int _index // à incrémenter / décrémenter pour savoir quel index initialiser

Class IConvoyorBelt: IConvoyorBelt.cpp/hpp

- methods:
  - ~IConvoyorBelt()
  - virtual void **put**(Object *) = 0
  - virtual Object ***take**() = 0 // renvoie l’objet présent
  - virtual **IN**() // reçoit des *Wraps* (avec un static int **voir juste en dessous**)
  // virtual car PapaXMasConvoyorBelt peut en avoir plusieurs
  - virtual **OUT**() // envoie les *Wraps* à Santa —> _on est **NULL**
- attributes (protected):
  - bool _occuped
  - Object *_on // set **NULL** au constructor

Le STATIC INT permettrait au premier tour de récupérer un *box*, ensuite un *GiftPaper* et ensuite
Un aléatoire entre un *Teddy* et *Littlepony*

//

Class PapaXmasTable: PapaXmasTable.cpp/hpp

- methods:
  - PapaXmasTable()
  - virtual void **put**(Object *) = 0
  - virtual Object ***take**() = 0 // renvoie l'objet présent
  - virtual Object ****look** = 0 // Retourne les objets présents (limite = 10 et le dernier est **NULL**)

Class PapaXmasConvoyorBelt: PapaXmasConvoyorBelt.cpp/hpp

- methods:
  - PapaXmasConvoyorBelt()
  - void **put**(Object *) = 0
  - Object ***take**() = 0 // renvoie l'objet présent
  - **IN**() // reçoit des *Wraps* en mode random
  - **OUT**()

//

ITable *createTable()
    return new ITable;

IConvoyorBelt *createConvoyorBelt()
    return new IConvoyorBelt;

##

/src/elf/

Class IElf : IElf.cpp/hpp

- methods:
  - sendGift() // Toutes les **ACTIONS** décrites après
- attributes (protected):
- attribute (public)
  - ITable *_table
  - IConvoyorBelt *_conB

ACTIONS:

This->_conB->IN() // le premier passage donne un *Box*
Object *obj = this->_conB->take()
Afficher « whistles while working » // il vient de faire un *take a wrap*
this->_table->put(obj)

// 2eme passage
This->_conB->IN() // donne un *GiftPaper*
Object *obj = this->_conB->take()
Afficher « whistles while working » // il vient de faire un *take a wrap*

this->_table->put(obj)

voir feuille + **AFFICHER « tuuuut tuuut tuut » au ‘wrapMeThat’**