Université d'Angers	Département Informatique
Faculté des Sciences	M1 Informatique, 06/07
Module : *Génie Logiciel, Modélisation, C++*	JEAN-MICHEL RICHER

Examen

(Aucun doucment autorisé)

Exercice 1 -Question ouverte (8 pts) - Votre employeur vous a confié la direction d'une équipe de 4 développeurs pour un projet d'une durée de 8 mois et pour lequel un budget prévisionnel a été établi. A la fin du projet, quel(s) document(s) allez vous produire et que contiendra/ont ces documents.

Exercice 2 -Question ouverte (4 pts) - Soit le planning suivant composé d'un ensemble de tâches dont la durée est exprimée en jours :

Tâche	Descriptif	Durée	Prédécesseurs
A	réviser le cours de GL	3	-
B	réviser le cours de Prolog	2	-
C	apprendre à programmer en C++	4	-
D	programmer Planner en Prolog	3	A, B
E	passer l'examen de GL avec succès	4	B, C

- dessiner le réseau PERT
- calculer les dates au plus tôt
- calculer les dates au plus tard
- indiquer le chemin critique

Exercice 3 -Modélisation C++ (8 pts) - On désire disposer de deux implantations pour gérer les listes d'objets :

- une implantation sous forme de tableau `class ArrayList`. Le tableau pourra être réalloué dynamiquement si on dépasse la taille initiale.
- une implantation sous forme de liste doublement chaînée `class LinkedList`

Les deux implantations possèdent les mêmes opérations de manipulation (`insert, get, remove,` ...) mais éventuellement avec des noms de méthodes différents. Par exemple l'insertion en tête de liste pourra s'appeler `insert_front` pour l'implantation en tableau et `push_front` pour l'implantation de liste chaînée :

```
class Object { ... };

class ArrayList {
  protected;
     int size, max_size;
     Object **tab;
  public:
    // constructeur avec taille maximale de la liste
    ArrayList(int n);
    // destructeur
    ~ArrayList();
    // insertion en debut de liste d'un object
    void insert_front(Object *obj);
    // acces a l'objet situe a la position n dans la liste
    Object *get_at(int n);
    // supprime l'objet en position n dans la liste et retourne
    // un pointeur sur cet objet
```

```
    Object *remove_at(int n);
     ...
};

class LinkedList {
  protected:
    int size;
    Link *first;
  public:
    // constructeur
    LinkedList();
    // destructeur
   ~LinkedList();
    // insertion en debut de liste d'un object
    void push_front(Object *obj);
    // acces a l'objet situe a la position n dans la liste
    Object *get(int n);
    // supprime l'objet en position n dans la liste et retourne
    // un pointeur sur cet objet
    Object *remove(int n);
    ...
};
```

On souhaite que le choix de l'implantation soit transparent pour l'utilisateur. Lorsque celui-ci crée une nouvelle liste, il spécifie uniquement le nombre N supposé d'éléments que contiendra la liste. Si $N <= 1000$, on utilisera une implantation sous forme de liste chainée. Par contre, si $N > 1000$, on prendra une implantation sous forme de tableau.

Proposez une solution à ce problème **en ne donnant que le code nécessaire à la compréhension de la solution**. Donnez éventuellement un diagramme de classes afin de représenter graphiquement la solution que vous envisagez.