**Exercice 1: 6 points** (1.5 + 2.5)

Dans l’exemple sous-dessous nous avons une file F et une pile P, chacune contient la représentation binaire du nombre entier 13 (1101).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **13** | **2** |  |  | |  | | --- | |  | | **1** | | **1** | | **0** | | **1** |   P |
| **1** | **6** | **2** |  |  |
|  | **0** | **3** | **2** |  |
|  |  | **1** | **1** | **2** |
|  |  |  | **1** | **0** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1** | **0** | **1** | **1** |  |  |

F

1. Écrire la procédure *binaire* (x, F)qui reçoit un nombre décimal *x* et crée une file F contenant sa représentation binaire.
2. Écrire la fonction *egale* (F, P) permettant de vérifier si la file F et la pile P contenant des représentations binaires pour le même nombre entier.

**Exercice 2: 4 points** (2 + 2)

Écrire les opérations (procédures ou fonctions) suivantes sur les listes chainées d’entiers.

1. est\_union (L1, L2, L3) permettant de vérifier si la liste L3 et l’union des deux listes L1 et L2.

**Exemple** : soit L1={2, 6, 12} et L2 = {7, 12}

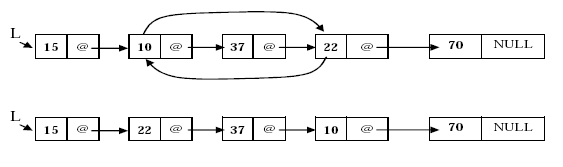
si L3={2,6,7} alors L3≠ L1 U L2

si L3={2, 7, 6,12} alors L3 = L1 U L2

si L3={2,6, 7, 12, 13} alors L3≠ L1 U L2

1. permutation (L, r1, r2) permettant de faire la permutation entre l’élément du rang r1 et l’élément du rang r2 en supposant que r1 et r2 sont inférieurs ou égale à la langueur du L.

**Exemple** : r1=2 et r2=4



**Exercice 3: 5 points** (1 + 2 + 2)

Une superette désire disposer d'un outil automatique pour traiter les informations concernant son stock. On vous propose de représenter ces informations sous forme d’une liste chainée où chaque maillon contient un produit représenté par sa référence (entier), sa désignation (chaine de caractères), son prix unitaire (réel) et la quantité disponible (réel).

1. Définir les types de données permettant de représenter ce stock.
2. Écrire la procédure Vendre (L, Ref, Quantite) permettant de retirer, si possible, la quantité 'Quantité’ du produit qui a pour référence **'**Ref**'.** Si la quantité est insuffisante ou le produit n’existe pas la procédure doit afficher "vente impossible".
3. Écrire la fonction récursive supprimer\_produits (L1, L2) permettant de supprimer de la liste L1 les produits qui existent dans la liste L2, sachant que chaque produit est identifié par sa référence.

**Exercice 4: 7 points** (0.5 + 3.5 + 1 + 2)

Nous considérons maintenant que les produits sont stockés dans un arbre binaire où chaque nœud de l’arbre contient un produit.

1. Donner la déclaration de cet arbre.
2. Écrire deux versions différentes pour la procédure Vendre de l’exercice précédente, la première en utilisant un arbre binaire simple et la deuxième en utilisant un arbre binaire de recherche.
3. Quelle est le nombre maximal des nœuds parcourus pour chacune des deux procédures précédentes.
4. Écrire la fonction meme\_contenu (a1, a2) permettant de vérifier si les deux arbres binaire de recherche a1 et a2 représentant le même stocke.

**\*\*Bonne** **chance\*\***