Rapport de projet en LC

Année :2021/2022

MOUHAMADOU GUEYE **P31 2127** Groupe 1B L2 INFO SAT

I. Exercice 13

Pour l'exercice 13, il s'agit de faire en premier lieu un programme zip(L) qui pour tout élément e, remplace une suite de n ≥ 1 fois dans la liste L par deux maillons contenant n et e.

Exemple : [2 2 2 4 5 5 5 5 5] devient [3 2 1 4 6 5]

Pour cela nous avons procéder comme suit :

D'abord on s'occupe de la tête de la liste c'est-à-dire du premier élément de la liste tout en prenant compte de tous les cas éventuellement possibles.

Ensuite pour le reste, on déclare deux pointeurs de listes qui vont assurer la suppression des maillons et l'insertion du nbre d'élément en le positionnant devant l'élément.

Enfin on retourne la tête de la liste ainsi obtenue.

Veuillez regarder le ' Programme zip(L) ' pour plus de clarté.

```
liste zip(liste l)
{
    liste c, pre,s,m;
    int n;
    if(estvide(l))
    {
        puts("liste vide ");
        return l;
    }
    else
    {

        if(!estvide(l))
        {
         n=1;
           if(!estvide(l->suiv))
        {
           while(!estvide(l->suiv) && l->done == (l->suiv)->done)
        {
           l = supp_tete(l);
           n++;
        }
        }
        l = inserer_tete(l,n);
        }
        c = l->suiv;
        s=c->suiv;
       while(!estvide(s))
        {
           n = 1;
           while(!estvide(s->suiv) && s->done==(s->suiv)->done)
        {
           c->suiv = s->suiv;
           s = detruir_maillon(s);
           s = c->suiv;
           n++;
        }
        m = creer_maillon(n);
        c->suiv = m;
        m->suiv = s;
        c = s;
        s = s->suiv;
        }
        return l;
    }
}
```

Programme zip(L)

En deuxième lieu, il s'agit de réaliser un programme unzip(L) qui fait l'action inverse du programme zip(L).

Pour y arriver, nous avons appliqué la même démarche que pour zip(L).

C'est-à-dire de s'occuper de la tête d'abord puis de s'occuper du reste en déclarant deux pointeurs de liste et enfin retourner la liste ainsi obtenue.

Veuillez regarder le ' Programme unzip(L) ' pour plus de clarté.

```
liste unzip(liste l)
{
    liste c = l->suiv,temp, s=c->suiv, m;
    int n;
    donne k;
    if(estvide(l))
    {
        puts("liste vide ");
        return l;
    }
    else
    {
       if(!estvide(l))
       {
        n = l->done;
        k = l->suiv->done;
        if(n==1)
        {
           l = supp_tete(l);
        }
        else
        {
           l = supp_tete(l);
           for (int i = 1; i < n; i++)
           l = inserer_tete(l, k);
        }
        while(!estvide(s))
        {
           n = s->done;
           k = s->suiv->done;
           c->suiv = s->suiv;
           s=detruir_maillon(s);
           s = c->suiv;
           // k = s->done;
           for (int i = 1; i < n;i++)
           {
               m = creer_maillon(s->done);
               c->suiv = m;
               m->suiv = s;
               c = m;
           }
           c = c->suiv;
           s = s->suiv;
        }
       }
       return l;
    }
```

Programme unzip(L)

II. Exercice 14

Pour l'exercice 14, il s'agit de réaliser 3 fonctions :

- afficher les éléments d'une liste l de type pPersonne :

```
void affiche_personne(pPersonne p)
{
    printf("PERSONNE =>  \n");

    printf("            Matricule =>>   %s \n",p->matr);
    printf("            Prenom    =>>   %s \n",p->prenom);
    printf("            Nom       =>>   %s \n",p->nom);
    printf("            Age       =>>   %d ans \n",p->age);
    puts("");
}
```

```
void affiche_liste_pers(pPersonne p)
{
    pPersonne c = p;
    if(estVide(p))
    {
        puts("liste VIDE !");
        erreur();
    }
    while (!estVide(c))
    {
        affiche_personne(c);
        c=c->suiv;
    }

}
```

- calculer la taille d'une liste l (le nombre d'éléments de la liste) :

```
int taille_liste(pPersonne p)
{
    int n =0;
    pPersonne c = p;
    while (!estVide(c))
    {
        n++;
        c = c->suiv;
    }
    return n;
}
```

- rechercher une personne par son matricule dans une liste l et retourner sa position :

```
int recherche(pPersonne p ,ch15 matricule)
{
    int pos = 1;
    pPersonne c = p;
    while(!estVide(c) && strcmpi(c->matr,matricule)!=0)
        {
            c = c->suiv;
            pos++;
        }
        if(estVide(c))
            return 0;
        else
        return pos;
}
```

III. Exercice 15

Pour l'exercice 15, il s'agit de réaliser 4 fonctions :

- insérer une personne à sa bonne place, suivant le nom, dans une liste triée :

```
pPersonne inserer_position_name(pPersonne p,pers x)
{
    pPersonne cour=p, pers = creer_personne(x);

    if(estVide(p))
        p = pers;
    else
    {
        if( strcmpi(cour->nom,pers->nom)>0)
            p=inserer_tete(p,x);
        else
        {
            while(!estVide(cour->suiv) && strcmpi(cour->suiv->nom,pers->nom)<0)
            {
                // pre = cour;
                cour = cour->suiv;
            }
            if(!estVide(cour->suiv))
            {
                pPersonne k = cour->suiv;
                cour->suiv = pers;
                pers->suiv = k;

            }
            else
                p = inserer_queue(p, x);
        }
    }
    return p;
}
```

- saisir une liste chainée de n personnes en respectant, au fur et mesure, l'ordre suivant le nom :

```
pPersonne remplir_liste_position_name(pPersonne p,int n)
{
    pers x;
    for (int i = 0; i < n;i++)
    {
        x = creer_pers(x);
        p = inserer_position_name(p,x);
    }

    return p ;
}
```

- fusionner deux listes chainées triées, dans une nouvelle liste, en respectant l'ordre :

```
pPersonne fusion_liste_personne(pPersonne p1,pPersonne p2)
{
    pPersonne p=p1;
    pers x;
    while(!estVide(p2))
    {
        x.ages = p2->age;
        strcpy(x.matr,p2->matr);
        strcpy(x.nom,p2->nom);
        strcpy(x.prenom,p2->prenom);

        p = inserer_position_name(p,x);
        p2 = p2->suiv;
    }

    return p;
}

pPersonne supprime_tete(pPersonne p)
{
    pPersonne c = p;
    p = p->suiv;
    c=supp_personne(c);
    return p;
}
```

- supprimer toutes les personnes âgées de plus de 50 ans :

```
pPersonne supp_personne_age(pPersonne p,int age)
{
    pPersonne c, s;
    while(!estVide(p) && (p->age==age))
        p = supprime_tete(p);
    if(!estVide(p))
    {
        c = p;
        s = p->suiv;
        while(!estVide(s))
        {
            if(s->age==age)
            {
                c->suiv = s->suiv;
                s = supp_personne(s);
                s = c->suiv;
            }
            else
            {
                c = s;
                s = s->suiv;
            }
        }
    }
    return p;
}
```