TP

Implémentation

Exercice 1

Dans `main.c`, définissez les trois constates suivantes, à utiliser dans le reste du TP :

- `LONGUEUR_MAX_MOT` : longueur maximale d'un mot (valeur laissée à votre choix)
- `NB_MAX_ETATS` : nombre maximal d'états autorisé pour un automate (utilisez une valeur supérieure ou égale à pour ce TP) ;
- `TAILLE_ALPHABET` : nombre de lettres dans l'alphabet (utilisez 2, puisque notre alphabet est toujours {*a*, *b*} dans ce TP).

```
#define LONGUEUR_MAX_MOT 100
#define NB_MAX_ETATS 5
#define TAILLE_ALPHABET 2
```

Exercice 2

On représente un automate en mémoire à l'aide d'une structure contenant champs :

- `nb_etats` : nombre d'états de l'automate ;
- `etat_initial` : numéro de l'état initial ;
- `etat_acceptant` : numéro de l'état acceptant ;
- `matrice_transition` : matrice de transition.

Dans `main.c`, définissez un type structuré appelé `t_automate`, qui contient ces champs

```
typedef struct
{ int nb_etats;
int etat_initial;
int etat_acceptant;
int matrice_transition[NB_MAX_ETATS][TAILLE_ALPHABET];
} t_automate;
```

Exercice 3

Écrivez la fonction `void affiche_automate(t_automate a)` qui affiche les champs de l'automate a passé en paramètre.

```
void affiche_automate(t_automate a)
{ int i,j;
printf("Le nombre d'etats de l'automate est %d\n", a.nb_etats);
printf("L'etat initial est E%d\n", a.etat_initial);
printf("L'etat acceptant est E%d\n", a.etat_acceptant);
printf("La matrice de transition est :\n ");
for(i=0;i<TAILLE_ALPHABET;i++)
printf("\t%c",'a'+i);
printf("\n");
for(i=0;i<a.nb_etats;i++)
{ printf("E%d",i);
for(j=0;j<TAILLE_ALPHABET;j++)
printf("\tE%d", a.matrice_transition[i][j]);
printf("\n");
} }
```

Exemple : si l'automate passé en argument est celui de l'exemple précédent, on obtiendra *exactement* l'affichage ci-dessous :

```
Le nombre d'etats de l'automate est 4
L'etat initial est E0
L'etat acceptant est E2
La matrice de transition est :
a b
E0 E1 E3
E1 E2 E1
E2 E2 E1
E3 E3 E3
```

Exercice 4

Écrivez la fonction `t_automate saisis_automate(t_automate a)` qui demande à l'utilisateur de saisir les différents champs constituant un automate. Les valeurs saisies sont utilisées pour initialiser l'automate passé en paramètre.

```
t_automate saisis_automate(t_automate a)
{
    int i,j;
    printf("Entrez SVP le nombrre d'etat : ");
    scanf("%d",&a.nb_etats);

    printf("Entrez SVP l'etat initial: ");
    scanf("%d",&a.etat_initial);

    printf("Entrez SVP l'etat acceptant : ");
    scanf("%d",&a.etat_acceptant );

    printf("Entrez SVP les transitions de la matrice :\n ");
    for (i=0; i<a.nb_etats; i++){
        printf("- Entrez les transitions de l'etat E%d :\n",i);
        for(j=0; j<TAILLE_ALPHABET; j++)
    {printf(" %c : ",'a'+j);
            scanf("%d",&a.matrice_de_transition[i][j]);

    }
    }
            return a;
}
```

Exercice 5

Écrivez une fonction `int applique_transition(t_automate a, int etat_courant, char lettre)`, qui reçoit en paramètres l'état courant, i.e. l'état actuellement occupé, et une lettre de l'alphabet A. La fonction calcule et retourne l'entier correspondant à l'état obtenu en suivant le lien associé à la lettre à partir de l'état courant, tel que décrit dans la matrice de transition de l'automate.

Remarque : cette fonction s'appelle la *fonction de transition* de l'automate. Son implémentation tient en une seule ligne.

Exemples :

- L'appel `applique_transition(a,1,'a')` renvoie la valeur 2.
- L'appel `applique_transition(a,1,'b')` renvoie la valeur 1.
- L'appel `applique_transition(a,2,'b')` renvoie la valeur 1.
- L'appel `applique_transition(a,3,'b')` renvoie la valeur 3.

```
int applique_transition(t_automate a, int etat_courant, char lettre)
{
    int resultat = a.matrice_de_transition[etat_courant][lettre-'a'];
    return resultat;
   }
```

Exercice 6

Écrivez une fonction `void teste_mot(t_automate a)` qui réalise les opérations suivantes :

1. Afficher l'automate reçu en paramètre ;
2. Demander à l'utilisateur de saisir une chaine de caractères contenant seulement des lettres de l'alphabet ;
3. Tester si le mot représenté par cette chaine de caractères est accepté ou pas par l'automate.
4. Afficher le résultat de ce test.

Testez votre programme avec l'automate donné en exemple.

- Fonction `teste_mot`

```
void teste_mot(t_automate a){
    char mot[LONGUEUR_MAX_MOT];
    char lettre_courante;
    int etat_courant;

    // on affiche l'automate recu
    affiche_automate(a);

    //initialiser le mot à 0
    int i;
    for(i=0; i<LONGUEUR_MAX_MOT; i++)
    {
        mot[i] = 0;
    }

    // on saisit le mot
    int longueur_mot ;
    printf("Entrez SVP la longueur de votre mot (<=100) : ");
    scanf("%d",&longueur_mot);

    printf("Entrez un mot composé de %d charactere contenant uniquement les
lettres a et b (entrez lettre apres lettre):\n ", longueur_mot);

    i=0;
    do{
       printf("Entrez la %d lette : ",i+1);
       scanf(" %c",&mot[i]);
       printf("\n");
       i++;
    }while(i<longueur_mot);

    etat_courant = a.etat_initial;

   for(i=0; i<longueur_mot; i++){
        lettre_courante = mot[i];
```

```
        etat_courant = applique_transition(a,etat_courant,lettre_courante);
    }

// on affiche le resultat
if(etat_courant==a.etat_acceptant)
    printf("Le mot \"%s\" est accepte par cet automate\n",mot);
else
    printf("Le mot \"%s\" est refuse par cet automate\n",mot);
}// la fin de la fonction teste
```

- Fonction `main` :

```
int main()
{
    // initialisation des valeurs de l'automoate exemple donnédans le TP
    t_automate a ={4, 0, 2, {{1,3},{2,1},{2,1}, {3,3}} };

    t_automate aa;
    aa = saisis_automate(aa);

    //appel de la fonction afficherr automate
    affiche_automate(aa);

    // appel de la transition applique etat
    int ec;
printf("Enrez SVP l'etat courant : ");
scanf("%d",&ec);
printf("\n");

    char l;
printf("Enrez SVP la lettre de l'alphabet : ");
scanf(" %c",&l);
    int e = applique_transition(aa,ec,l);
    printf("%d",e);
    printf("\n");

    // Appel de la fonction test mot
    teste_mot(aa);
    return 0;
}
```