INF203 - Exercices semaine 9

Automates

Exercice 1 :

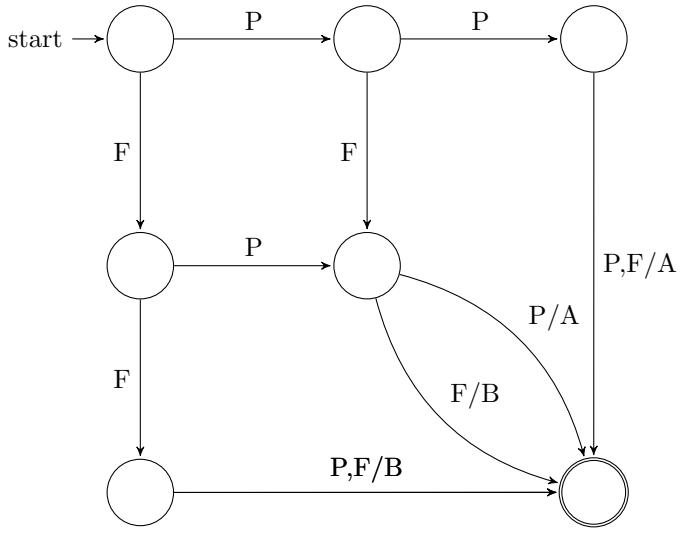
Anémone et Barnabé jouent à pile ou face. Anémone marque un point quand la pièce tombe sur pile, Barnabé marque un point quand la pièce tombe sur face. En fonction de leur humeur, ils jouent selon l'une des trois règles suivantes :

1. ils lancent la pièce 3 fois exactement. Celui qui a le plus de points gagne la partie.
2. ils jouent tant que l'un des joueurs n'a pas 2 points de plus que l'autre. Dès que l'un des joueurs a 2 points de plus que l'autre, il gagne la partie.
3. ils jouent tant que l'un des joueurs n'a pas 2 points de plus que l'autre, ou qu'ils n'ont pas lancé la pièce 6 fois. Si l'un des joueurs a 2 points de plus que l'autre, il gagne la partie. Sinon, c'est match nul.

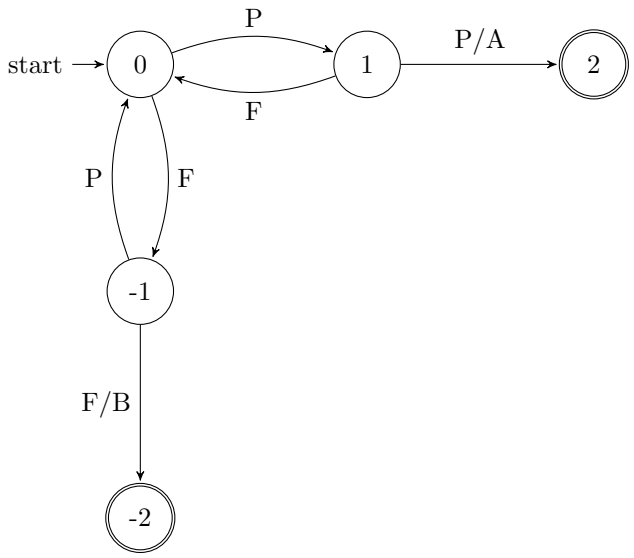
Dessiner les automates modélisant chacune des 3 règles du jeu.

Solution :

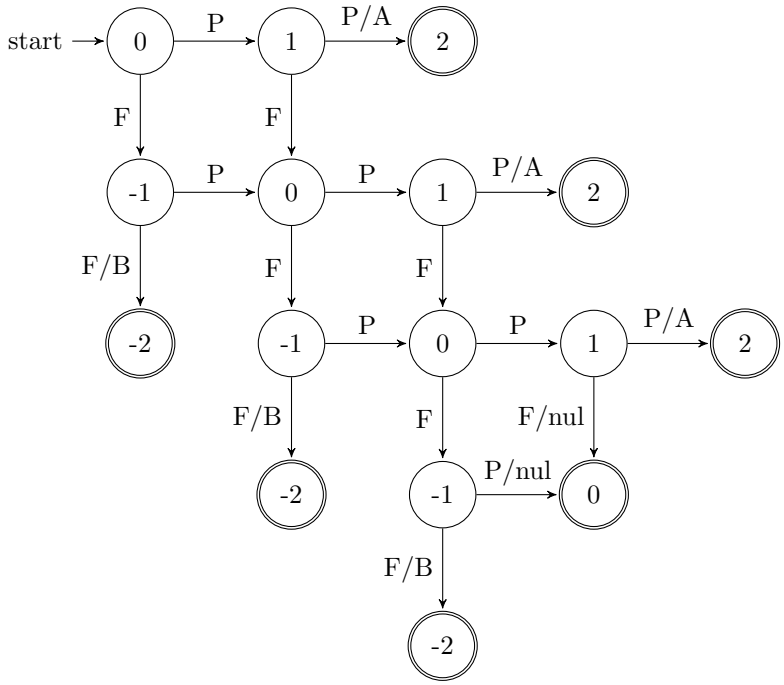
1. lorsque seuls trois lancers sont effectués :



2. lorsque le jeu continue jusqu'à un écart de deux points : on regarde les scores du point de vue d'Anémone,



3. lorsque le jeu se limite à un maximum de 6 coups ou à un écart de 2 points, on doit répliquer l'automate précédent en dupliquant l'état 0 au moins 2 fois :



Exercice 2 :

Anémone et Barnabé jouent maintenant au tennis. Comme ils sont fatigués, ils décident de ne jouer qu'un jeu. La règle pour gagner un jeu est de gagner au moins 4 points et d'avoir au moins deux points de plus que l'adversaire. Le jeu s'arrête dès que cette condition est réalisée par l'un des deux joueurs. Soit pt_A et pt_B le nombre de points marqués par Anémone et Barnabé. Voici les scores possibles en fonctions de pt_A et pt_B.

$pt_A = 0$ **et** $pt_B = 0$ donne le score $0 - 0$.

$pt_A = 1$ **et** $pt_B = 0$ donne le score $15 - 0$.

$pt_A = 0$ **et** $pt_B = 1$ donne le score $0 - 15$.

$pt_A = 1$ **et** $pt_B = 1$ donne le score $15 - 15$.

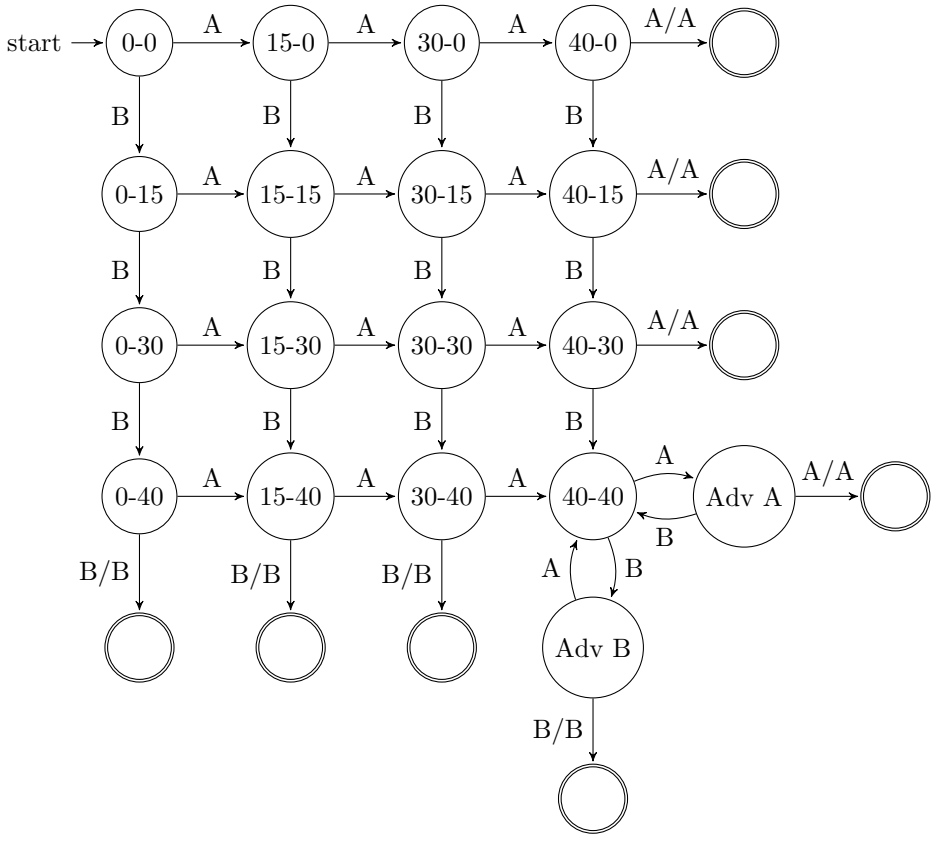
$pt_A = 2$ **et** $pt_B = 0$ donne le score $30 - 0$.

$pt_A = 0$ **et** $pt_B = 2$ donne le score $0 - 30$.

$pt_A = 2$ **et** $pt_B = 1$ donne le score $30 - 15$.
$pt_A = 1$ **et** $pt_B = 2$ donne le score $15 - 30$.
$pt_A = 2$ **et** $pt_B = 2$ donne le score $30 - 30$.
$pt_A = 3$ **et** $pt_B = 0$ donne le score $40 - 0$.
$pt_A = 0$ **et** $pt_B = 3$ donne le score $0 - 40$.
$pt_A = 3$ **et** $pt_B = 1$ donne le score $40 - 15$.
$pt_A = 1$ **et** $pt_B = 3$ donne le score $15 - 40$.
$pt_A = 3$ **et** $pt_B = 2$ donne le score $40 - 30$.
$pt_A = 2$ **et** $pt_B = 3$ donne le score $30 - 40$.
$pt_A \geq 3$ **et** $pt_A = pt_B$ donne le score $40 - 40$.
$pt_A \geq 3$ **et** $pt_A = pt_B + 1$ donne « Avantage A(némone) ».
$pt_B \geq 3$ **et** $pt_B = pt_A + 1$ donne « Avantage B(arnabé) ».
$pt_A \geq 4$ **et** $pt_A \geq pt_B + 2$ donne « Jeu A(némone) ».
$pt_B \geq 4$ **et** $pt_B \geq pt_A + 2$ donne « Jeu B(arnabé) ».

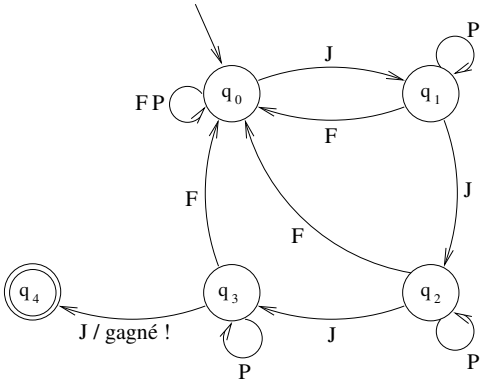
Dessiner l'automate modélisant un jeu de tennis entre Anémone et Barnabé.

Solution :



Exercice 3 :

Dans un jeu télévisé, on pose à un candidat une série de questions ; il n'a droit qu'à une seule réponse par question. L'automate ci-dessous modélise le déroulement du jeu en fonction des réponses du candidat : J pour "réponse juste", F pour "réponse fausse" et P pour "je passe".



Rédigez en français de façon claire et sans ambiguïté les règles de ce jeu.

Solution :

C'est une variante du "4 à la suite" de "Questions pour un Champion". Il s'agit de répondre correctement à une question 4 fois de suite. Chaque erreur nous renvoie au point de départ. Il est par contre possible de "passer" une question qui ne nous convient pas sans retomber à 0, ce qui n'est pas le cas dans le jeu télévisé.

Exercice 4 :

entrées →

états ↓ trans	0	1	2	3
0	1	2	2	3
1	1	2	2	3
2	1	2	2	3
3	3	3	3	3

entrées →

états ↓ sortie	0	1	2	3
0				
1		bip	bip	bip
2				
3				

Les tableaux ci-dessus représentent la fonction de transition et la fonction de sortie d'un automate A. L'état initial est 0, et il y a un seul état final, l'état 3.

Q1. Dessinez l'automate A.
Les entrées de cet automate correspondent à des catégories de caractères :
- les lettres (majuscules ou minuscules) sont codées par l'entier 0,
- les symboles de ponctuation sont codées par l'entier 1,
- les séparateurs (espaces, tabulations ...) sont codées par l'entier 2,
- la fin de fichier est codée par l'entier 3.

Reportez cette information sur le dessin de votre automate (notez par exemple `L` pour lettre, `P` pour ponctuation, `E` pour espace, `EOF` pour la fin de fichier).
Q2. Simulation de l'automate à la main. On initialise une variable entière `x` à `0`, et on lance la simulation de l'automate avec la séquence d'entrée suivante :

```
Et hop !!! On lance la simulation ... et on compte les bips !
```

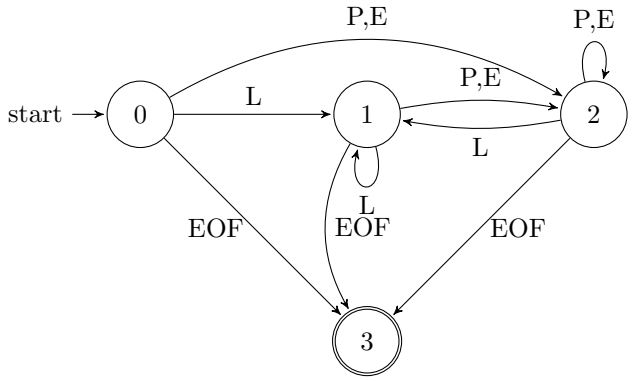
À chaque `bip`, la variable `x` est incrémentée de `1`.
- lorsqu'on arrive dans l'état final (`3`), combien vaut `x` ?
- que représente `x` par rapport au texte d'entrée ?

Q3. Programme de simulation de l'automate.
En utilisant les fonctions et déclarations fournies, complétez le programme suivant (parties `/* -> A COMPLETER */` de la fonction `main`) pour simuler l'automate A à partir d'une séquence d'entrée lue dans un fichier de nom donné en argument de la ligne de commande :

```c
#include <stdio.h>
#include <ctype.h>
#define LETTRE 0
#define PONCTUATION 1
#define ESPACE 2
#define FIN_FICHIER 3

int transition[4][4] = {
{1, 2, 2, 3},
{1, 2, 2, 3},
{1, 2, 2, 3},
{3, 3, 3, 3}
};

int lire_nature_entree(FILE * f) {
    char c;
    int nature;
    fscanf(f, "%c", &c);
    if (feof(f))
        nature = FIN_FICHIER;
    else if (isalpha(c))
        nature = LETTRE;
    else if (isspace(c))
        nature = ESPACE;
    else
        nature = PONCTUATION;
    return nature;
}

int sortie(int etat, int nature_entree) {
    int bip = 0;
    if ((etat == 1) && (nature_entree != LETTRE))
        bip = 1;
    return bip;
}


int main(int argc, char *argv[]) {
    FILE *f;
    int x;
    int etat_courant, etat_suivant;
    int nature_entree;

    /* Vérification du nombre d'arguments, ouverture du fichier avec vérification */
    /* -> A COMPLETER */

    /* Simulation */
    /* -> A COMPLETER */

    printf("x = %d\n", x);
    return 0;
}
```

Solution :

Q1. L'automate peut être représenté comme suit :



Q2. On remarque que chaque `bip` intervient lorsque l'on sort de l'état 1 qui correspond à une série de lettres. Les occurrences de `bip` correspondent donc au nombre de mots composant l'entrée. Ici on trouvera donc que `x=11`.

Q3.
```
#include <stdio.h>
#include <ctype.h>
#define LETTRE 0
#define PONCTUATION 1
#define ESPACE 2
#define FIN_FICHIER 3

int transition [4][4]={
{1 , 2 , 2 , 3} ,
{1 , 2 , 2 , 3} ,
{1 , 2 , 2 , 3} ,
{3 , 3 , 3 , 3}
};

int lire_nature_entree (FILE * f) {
    char c ;
    int nature ;
    fscanf (f,"%c",&c);
    if (feof(f))
        nature = FIN_FICHIER;
    else if (isalpha(c))
        nature = LETTRE;
    else if (isspace(c))
        nature = ESPACE;
    else nature = PONCTUATION;

    return nature ;
}

int sortie ( int etat , int nature_entree ) {
    int bip = 0;
    if (( etat == 1) && ( nature_entree != LETTRE ))
        bip = 1;
    return bip ;
}

int main ( int argc , char * argv []) {
    FILE * f ;
    int x=0 ;
    int etat_courant=0, etat_suivant ;
    int nature_entree ;

    /* Vérification du nombre d'arguments , ouverture du fichier avec vérification */
    f = fopen(argv[1],"r");
```

```
        if (f == NULL){
            printf("Fichier inexistant");
            return 1;
        }

        /* Simulation */
        nature_entree=lire_nature_entree(f);
        while(nature_entree!=FIN_FICHIER){
            etat_suivant=transition[etat_courant][nature_entree];
            x += sortie(etat_courant,nature_entree);

            printf("Etat courant = %d\n",etat_courant);
            etat_courant=etat_suivant;
            printf("> Etat courant = %d\n",etat_courant);
            nature_entree=lire_nature_entree(f);
        }



        printf("x=%d\n",x);
        return 0;
    }
```

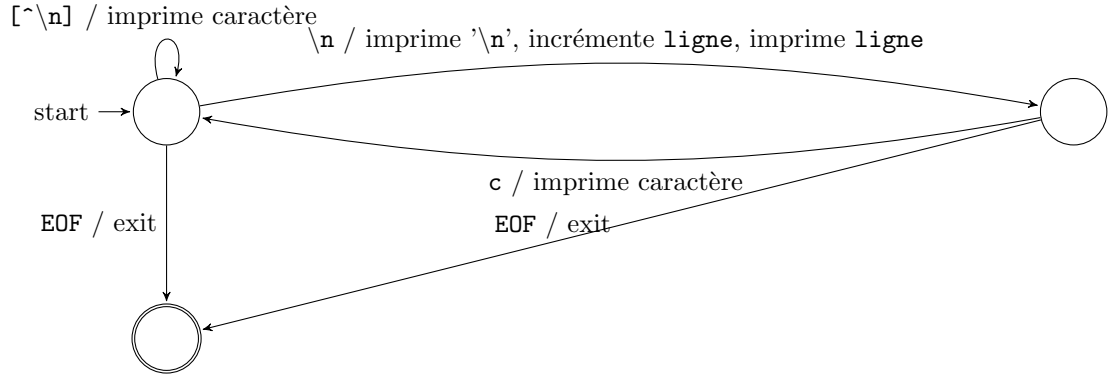
Exercice 5 :

Écrire un programme qui lit un fichier et le ré-affiche à l'écran (ou dans un autre fichier selon le nombre d'arguments donnés) en précédant chaque ligne par son numéro.

Au préalable. Modéliser le problème à l'aide d'un automate et d'un compteur de lignes. Les entrées de l'automate sont les caractères lus. Ses sorties sont les caractères et les entiers écrits, et/ou l'incrémentation du compteur.

Solution :

Un automate possible est le suivant :



et le code associé :

```
#include <stdio.h>

int main (int argc,char *argv []) {
    FILE *f;

    f = fopen(argv[1],"r");
    if (f == NULL){
        printf("Fichier inexistant");
        return 1;
    }
```

```
    char c;
    int lignes = 1;
    printf("1. ");
    fscanf(f,"%c",&c);
    while(!feof(f)){
        printf("%c",c);
        if (c=='\n')
            printf("%d. ",++lignes);
        fscanf(f,"%c",&c);
    }

    return 0;
}
```