Exercices de Langage C et Programmation Système

1. Les instructions de base en langage C

Exercice 1.1. Quels affichages ?

```c
#include<stdio.h>
int main() {
  char c = 'E';
  printf("\tAffichage d'un caractere\n");
  printf("%%c -> %c\n", c);
  printf("%%d -> %d\n", c);
  printf("%%x -> %x\n", c);

  int e = 72;
  printf("\tAffichage d'un entier\n");
  printf("%%c -> %c\n", e);
  printf("%%d -> %d\n", e);
  printf("%%x -> %x\n", e);

  float f = 19.6F/100;
  printf("\tAffichage d'un nombre reel\n");
  printf("%%f -> %f\n", f);
  printf("%%.3f -> %.3f\n", f);
  printf("%%.2f -> %.2f\n", f);

  printf("\tAffichage d'un texte\n");
  printf("%%s -> %s\n", "3iL ingé");
  printf("%%.3s -> %.3s\n", "3iL ingé");
  printf("bonjour %%-*.*s! -> bonjour %-*.*s!\n", 4, 2, "Julie");
  return 0;
}
```

Exercice 1.2. Division euclidienne

Écrire un programme qui calcul affiche le résultat et le reste de la division entière de deux valeurs saisies par l'utilisateur (short).

Exemple d'affichages et saisies :

Saisissez le dividende
370
Saisissez le diviseur
24
370 / 24 = 15 et reste 10

Exercice 1.3. Les opérateurs

Qu'affiche le programme suivant ?

```
#include<stdio.h>
int main() {
  int e = 72;
  printf("%d\n", e != 72);
  printf("%d\n", e && 72);
  printf("%d\n", !e || 0 && e);
  printf("%d\n", e & 123);
  printf("%d\n", -~e | 23);
  printf("%d\n", e ^ 23);
  printf("%d\n", e<<2);
  printf("%d\n", e>>2);
  return 0;
}
```

Exercice 1.4. Dans le livre

Livre "Langage C" (2nd édition) de Frédéric DROUILLON – éditions ENI – juillet 2021

Livre Langage C – Chapitre II – 4

Livre Langage C – Chapitre III – 8

Livre Langage C – Chapitre II – 6

Livre Langage C – Chapitre III – 4 et 6

Livre Langage C – Chapitre IV – 4

2. Les instructions conditionnelles

Exercice 2.1. Calculatrice 1

Créer une calculatrice simplifiée :

Affichage et saisies :

Saisissez le premier opérande
123
Saisissez l'opérateur (+ ou –)
+
Saisissez le second opérande
45
123 + 45 = 168

Exercice 2.2. Calculatrice 2

Que réalise le programme suivant ? Corriger les erreurs éventuelles

```
#include<stdio.h>
//Que fait-il ?
int main() {
  int op1, op2, res;
  char operateur;
  printf("saisissez le premier operande\n");
  scanf(" %d", &op1);
  printf("saisissez l'operateur (* ou /)\n");
  scanf(" %c", &operateur);
  printf("saisissez le second operande\n");
  scanf(" %d", &op2);
  if(operateur=='/')
    if(op2!=0)
      res = op1 / op2;
  else
    res = op1 * op2;
  printf("%d %c %d = %d\n", op1, operateur, op2, res);
  return 0;
}
```

Exercice 2.3. Calculatrice 3

Modifier le programme de calculatrice pour pouvoir saisir l'opération en une seule fois :

Exemple d'affichages et saisies :

Saisissez l'opération :
123 – 45
123 – 45 = 78

Exercice 2.4. Calculatrice 4

Réécrire le programme de l'exercice précédent sans utiliser l'instruction **`if`**.

Exercice 2.5. Calculatrice 5

Réaliser une calculatrice permettant d'effectuer les opérations +, -, *, / et % avec l'instruction **`switch`**.

Exercice 2.6. Dans le livre

Livre Langage C – Chapitre IV – 2.6

Livre Langage C – Chapitre IV – 3.4

3. Les instructions itératives

Exercice 3.1. Le livret d'épargne

Soit un livret d'épargne avec un taux à 3%. Un épargnant indique la somme placée et un nombre d'années de placement. Afficher la somme obtenue à la fin de chaque année.

Exercice 3.2. La complexité d'un mot de passe

Faire saisir un mot de passe à l'utilisateur. Lui indiqué combien de caractères ont été saisis, s'il a utilisé des minuscules, des majuscules, des chiffres et des caractères spéciaux.

Un message annonce enfin la complexité du mot de passe en utilisant le tableau suivant :

Taille	*Chiffres uniquement*	*Lettres minuscules ou (excusif) majuscules*	*Lettres minuscules et majuscules*	*Chiffres et lettres minuscules et majuscules*	*Chiffres et lettres minuscules et majuscules et caractères spéciaux*
1 à 5	Mauvais	Mauvais	Mauvais	Mauvais	Mauvais
6	Mauvais	Mauvais	Mauvais	Trop simple	Trop simple
7	Mauvais	Mauvais	Trop simple	Trop simple	Trop simple
8	Mauvais	Trop simple	Trop simple	Trop simple	Trop simple
9	Mauvais	Trop simple	Trop simple	Simple	Simple
10	Mauvais	Trop simple	Simple	Simple	Simple
11	Trop simple	Simple	Simple	Bon	Bon
12	Trop simple	Simple	Bon	Bon	Bon
13	Trop simple	Simple	Bon	Bon	Très bon
14 et plus	Trop simple	Bon	Bon	Très bon	Très bon

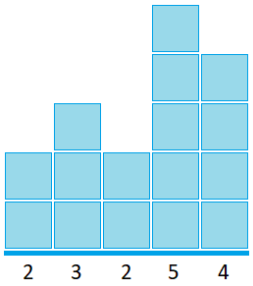
Exemple d'affichages et saisies :

Pa$$w0rd
il y a 8 caractères dont
- 1 chiffre
- 4 minuscules
- 1 majuscule
- 2 caractères spéciaux

Qualité du mot de passe : trop simple

Exercice 3.3. L'alpiniste (Examen 2024)

Un alpiniste prépare une excursion le long d'un trajet décrit par une séquence de nombres : les altitudes des étapes successives du trajet. Ainsi, la séquence de nombres "2 3 2 5 4" décrit le trajet illustré ci-dessous.



L'alpiniste met deux minutes pour chaque unité d'altitude qu'il doit gravir pour atteindre l'étape suivante, et une minute pour chaque unité d'altitude dont il doit descendre.

Écrivez un programme qui calcule le temps que met l'alpiniste pour aller de la première à la dernière étape.

Dans l'exemple ci-dessous, l'alpiniste part de l'altitude 2, met 2 minutes pour passer à l'altitude 3, puis 1 minute pour revenir à l'altitude 2, 6 minutes pour gravir jusqu'à l'altitude 5, et enfin 1 minute pour descendre à l'altitude 4.

Au total, l'alpiniste met donc 2 + 1 + 6 + 1 = 10 minutes.

Faire d'abord saisir le nombre d'altitudes (qui ne dépasse pas 100) puis les différentes altitudes. Enfin afficher le temps de parcours.

Exercice 3.4. Dans le livre

Livre Langage C – Chapitre IV – 5.6

4. Les instructions peu orthodoxes

Exercice 4.1. L'exorciste 1

Réécrire le code suivant sans l'instruction `break`.

```
int saisie;
for(int nbEssai=0; nbEssai<3; nbEssai++) {
  printf("code carte bleu ?\n");
  scanf(" %d", &saisie);
  if(saisie==PWD)
    break;
}
if(saisie==PWD)
  printf("Bon mot de passe\n");
else
  printf("Carte confisquee\n");
```

Exercice 4.2. L'exorciste 2

Réécrire le code suivant sans l'instruction `continue`.

```
int beuzeu = 7;
for(int i=1; i<100; i++) {
  if(i%beuzeu==0 || i%10==7 || i/10==7) {
    printf("Beuzeu\n");
    continue;
  }
  printf("%d\n", i);
}
```

Exercice 4.3. L'exorciste 3

Réécrire le code suivant sans l'instruction `goto`.

```
int main() {
  //...
  switch(operateur) {
  //...
  case '/':
    if(op2==0)
      goto erreurDiv;
    res = op1 / op2; break;
  default :
    goto erreurOp;
  }
  printf("%d %c %d = %d\n", op1, operateur, op2, res);
  return 0;
erreurDiv:
  printf("La division par 0 est interdite\n"); return 1;
erreurOp:
  printf("Operateur inconnu\n"); return 2;
}
```

Exercice 4.4. Dans le livre

5. Les tableaux

Exercice 5.1. Initialisation d'un tableau avec une boucle

Initialiser un tableau de dix cases avec les dix premières puissances de deux.

Réinitialiser le tableau avec les dix premiers nombres premiers.

Exercice 5.2. MAJUSCULES

Écrire un programme qui transforme un texte saisi par l'utilisateur tout en majuscule et l'affiche.

Même exercice pour uniquement la première lettre de chaque mot

Exercice 5.3. Palindrome

Un palindrome est une figure de style désignant un texte ou un mot dont l'ordre des lettres reste le même qu'on le lise de gauche à droite ou de droite à gauche.

Exemples : Ici, Kayak, Ésope reste ici et se repose, Karine alla en Irak, Engage le jeu que je le gagne...

Écrire un programme qui demande une chaine de caractères à l'utilisateur et lui indique si celle-ci est un palindrome.

Version 1 : L'utilisateur saisie son texte avec uniquement des caractères alphabétiques non accentués.

Version 2 : Les caractères alphabétiques accentués sont transformés en l'équivalent sans accent et les caractères non alphabétiques sont ignorés.

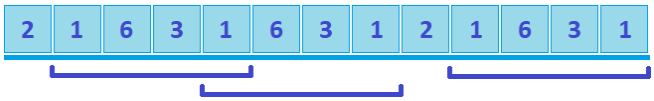
Exercice 5.4. Les blocs de 4 (Examen 2024)

Soit une liste de nombres saisis par l'utilisateur. Trouvez le bloc de 4 nombres successifs qui est présent le plus grand nombre de fois dans la liste.

Par exemple dans la liste ci-dessous, le bloc constitué des nombres 5 3 2 1 dans cet ordre, est présent deux fois. C'est le seul bloc présent plusieurs fois, donc c'est la réponse attendue.

Dans un deuxième exemple ci-dessous, c'est le bloc constitué des nombres 1 6 3 1 dans cet ordre, qui est présent le plus de fois (trois fois). Le bloc 2 1 6 3 est quant à lui présent deux fois seulement.

Notez que la deuxième copie du bloc 1 6 3 1 recouvre en partie sa première copie.



Demander à l'utilisateur de saisir le nombre de valeurs (100 au maximum) puis les différentes valeurs (valeur maximale de 100).

Afficher le nombre maximal d'occurrence d'un même bloc de 4 nombres successifs.

Information : pour éviter de saisir vos valeurs de test à chaque fois vous pouvez créer un fichier contenant ces valeurs et rediriger l'entrée standard depuis ce fichier :

```
$ ./exo5-4 < fichierTest5-4.txt
```

Avec `fichierTest5-4.txt` contenant par exemple :

```
10
4
5
3
2
1
6
5
3
2
1
```

Exercice 5.5. La micro bataille navale 1

Réaliser un algorithme permettant de jouer à la bataille navale avec les simplifications suivantes :

- Il n'y a qu'un seul joueur.
- Il n'y a que quatre lignes et quatre colonnes
- Il n'y a qu'un seul bateau qui n'occupe qu'une seule case (cette case est choisie aléatoirement)

Exemple d'affichages et saisies :

```
????
????
????
????
Quelle colonne ? 2
Quelle ligne ? 2
Plouf ! À l'eau !
????
?~??
????
????
Quelle colonne ? 4
Quelle ligne ? 1
Plouf ! À l'eau !
???~
?~??
????
????
Quelle colonne ? 1
Quelle ligne ? 3
Boom ! Touché coulé
Bravo, vous avez gagné !
```

Exercice 5.6. Le morpion

Créer un algorithme permettant de jouer au morpion : Dans une grille de trois cases par trois, les joueurs écrivent tour à tour leur symbole (X et O) dans une case vide. Le premier joueur qui parvient à aligner (en ligne, en colonne ou en diagonale) trois de ses symboles a gagné.

Exercice 5.7. Dans le livre

Livre Langage C – Chapitre VI – 1.7

Livre Langage C – Chapitre VI – 2.5

6. Les types évolués

Exercice 6.1. Le restaurant

Écrire un algorithme permettant de passer commande puis de l'afficher (en utilisant des énumérations).

- 3 Formules sont disponibles
  - entrée + plat
  - plat + dessert
  - entrée + plat + dessert

- Entrées
    - Salade de crudités
    - Pâté en croute
- Plats
    - Steak (choix de la cuisson : saignant, à point, bien cuit)
    - Poisson
- Dessert
    - Gâteau
    - Glace

Exercice 6.2. La date

Créer une structure date constitué d'un numéro pour le jour, d'une valeur d'une énumération pour le mois et d'un numéro pour l'année.

Faire saisir une date à l'utilisateur.

Indiquer :

- Si cette date est dans une année bissextile ;
- Si elle est correcte (le 31 novembre n'existe pas par exemple) ;
- La date décomposée en jour, mois en toutes lettres et année ;
- Le numéro du jour de l'année.

Exercice 6.3. Dans le livre

Livre Langage C – Chapitre V – 1.4

Livre Langage C – Chapitre V – 2.3

Livre Langage C – Chapitre VI – 3.4

7. Les pointeurs

Exercice 7.1. Manipulation d'une valeur à travers un pointeur

Déclarer une variable de type réel et un pointeur vers cette variable.

Incrémenter la valeur de la variable en passant par le pointeur.

Afficher la valeur, l'adresse mémoire pointée par le pointeur et l'adresse mémoire de la variable de type pointeur.

Exercice 7.2. Adresses des cases d'un tableau

Créer et initialiser un tableau d'entiers de 10 cases.

Afficher l'adresse mémoire de chaque case.

Faire de même avec un tableau de char, de short et de double.

Que remarquez-vous ?

Exercice 7.3. Occurrences

Créer une structure avec un champ entier nb et un champ caractère car.

Faire saisir un texte à l'utilisateur.

Indiquer le nombre de fois que chaque lettre a été saisie

Exemple :

bonjour -> b : 1 o : 2 n : 1 j : 1 u : 1 r : 1

Exercice 7.4. Dans le livre

Livre Langage C – Chapitre IX – 1.10.1

8. Les fonctions

Exercice 8.1. C'est le plus grand

Écrire une fonction prenant deux valeurs réelles en paramètres et qui retourne la plus grande des deux.

Écrire une autre fonction prenant également deux valeurs réelles en paramètres mais qui retourne

- 0 si les deux valeurs sont égales
- 1 si c'est la première valeur qui est la plus grande
- -1 sinon

Écrire un algorithme principal faisant appel à ces deux fonctions

Exercice 8.2. Le jeu du saute-mouton

L'objectif de ce jeu est de se faire se croiser deux troupeaux de moutons. Initialement les moutons sont dans la configuration suivante :

Au final, il faut arriver à cette configuration :

Pour y parvenir, il faut respecter les règles suivantes :

- Un mouton peut avancer d'un cran à condition que la place devant lui soit libre.
- Un mouton peut sauter un mouton qui va en sens inverse si la place suivante est libre.
- Aucun autre déplacement n'est disponible.

Les données sont représentées par un tableau comme ceci :

0	1	2	3	4	5	6
>	>	>		<	<	<

Exemple d'affichages et saisies :

```
 0 1 2 3 4 5 6
|>|>|>|  |<|<|<|
Quel pion voulez-vous déplacer?
2
 0 1 2 3 4 5 6
|>|>|  |>|<|<|<|
Quel pion voulez-vous déplacer?
```

…

Voici un plan d'action :

- Essayer de jouer (avec un crayon et un papier) , afin de comprendre le mécanisme du jeu, de maitriser les différents déplacements possibles ainsi que les situations de blocage.
- Rechercher les sous algorithmes nécessaires et déterminer s'il s'agit de procédures ou des fonctions.
- Écrire l'algorithme principal en supposant avoir à votre disposition les fonctions et procédures définies précédemment.
- Écrire les fonctions et procédures

Exercice 8.3. ASCII Art Studio 1

Écrire un algorithme permettant de réaliser des dessins en ASCII Art. Le dessin est modélisé par un tableau de caractères de dix lignes et de quarante colonnes. Initialement ce tableau ne contient que des caractères espace. Une procédure doit réaliser l'affichage de l'œuvre.

Au lancement du programme, le dessin est affiché et en dessous un menu propose de réaliser l'une des fonctionnalités suivantes :

- Écrire un caractère à l'emplacement désiré
- Dessiner un rectangle avec un ensemble de caractères de taille et de position choisies
- Changer "la couleur" (C'est-à-dire le caractère) en utilisant un pot de peinture

Chacune des fonctionnalités citées doit être réalisée à l'aide d'une ou plusieurs procédures.

Exercice 8.4. Dans le livre

Livre Langage C – Chapitre IV – 7.10

Livre Langage C – Chapitre VI – 4.5

Livre Langage C – Chapitre VIII – 8

9. L'allocation dynamique de mémoire

Exercice 9.1. Un tableau et des fonctions

Écrire une fonction qui crée un tableau à une dimension de dix entiers et initialise aléatoirement les valeurs de ce tableau avec des valeurs comprises entre une borne minimale et une borne maximale passées en paramètres.

Créer une fonction qui retourne la plus grande valeur d'un tableau (ce tableau ne fait pas nécessairement dix cases…).

Écrire un algorithme faisant appel à ces deux fonctions.

Exercice 9.2. La micro bataille navale 2

Reprendre l'algorithme de la bataille navale en le découpant en procédures et fonctions :

- Écrivez une procédure initialiserPlateau() qui initialise le plateau de jeu.
- Écrivez une procédure afficherPlateau() qui comme son nom l'indique affiche le plateau de jeu.
- Écrivez une fonction saisirEntierEntreBornes() qui fait saisir à l'utilisateur des valeurs jusqu'à ce que celle-ci soit comprise entre une valeur minimum et une valeur maximum passées en paramètre de cette fonction.
- Écrivez une fonction tirer() qui permet au joueur de choisir ses coordonnées de tir et retourne vrai si le bateau est touché et faux sinon.

Exercice 9.3. La micro bataille navale 3

Modifier le programme précédent pour que la taille du plateau de jeu soit décidée par le joueur.

Exercice 9.4. Dans le livre

Livre Langage C – Chapitre IX – 1.10

Livre Langage C – Chapitre IX – 2.7

Livre Langage C – Chapitre IX – 3.3

10. Les fichiers

Exercice 10.1. La liste de course

Écrire un programme permettant à un utilisateur de saisir des articles pour sa liste de courses. Ces articles sont enregistrés dans un fichier.

Exercice 10.2. ASCII Art Studio 2

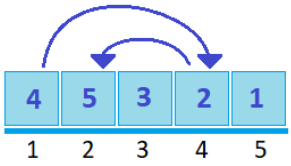
Ajouter les fonctions d'enregistrement et de chargement des images ASCII.

Exercice 10.3. Le parcours numéroté (Examen 2024)

Un parcours est constitué d'une suite de cases numérotées de 1 à nbCases.

Dans chaque case se trouve le numéro d'une autre case. Le parcours commence à la case numéro 1. À chaque étape, on lit le nombre contenu dans sa case actuelle, puis on se déplace sur la case dont le numéro est ce nombre.

Ainsi dans l'exemple suivant, on a 5 cases numérotées de 1 à 5. On commence le parcours dans la case de gauche. Cette case contient le nombre 4, donc pour l'étape suivante, on va à la case 4. La case 4 contient le numéro 2, donc pour l'étape suivante, on va à la case 2, et ainsi de suite.



Écrivez un programme qui lit la description d'une grille dans un fichier, puis affiche les numéros des 10 premières cases par lesquelles on passe lors du parcours. Notez que la même case peut être présente plusieurs fois dans ce parcours.

Structure du fichier :

La première ligne de l'entrée contient un entier nbCases, le nombre de cases de la grille, entre 1 et 100.

Les nbCases lignes suivantes contiennent chacune un entier, le contenu des cases de la grille, dans l'ordre.

Exemple :

```
5
4
5
3
2
1
```

Exercice 10.4. Dans le livre

Livre Langage C – Chapitre X – 7

11. Langage C et Linux

12. Les fichiers

Exercice 12.1. La création d'un fichier avec les droits souhaités

Créer un fichier dont le nom est passé sur la ligne de commande avec les droits choisis par l'utilisateur. Le contenu du fichier créé doit être le suivant :

```
# !/bin/bash
echo "Hello World !"
```

Exercice 12.2. Le compte est bon

Compter le nombre de caractères présents dans un fichier dont le nom est passé sur la ligne de commande.

Exercice 12.3.

13. Les processus

Exercice 13.1. Je suis ton père…

Écrire un programme se scindant en deux processus qui affichent chacun un message :

Je suis ton père
Je suis ton fils

Exercice 13.2. La course

Deux processus affichent chacun 1000 fois un caractère. Le premier à avoir terminé gagne.

Afficher le résultat, par exemple :

le processus 4587 a gagné (enfant 1)

Exercice 13.3. Le contenu des répertoires

Écrire un programme qui fait saisir à l'utilisateur le chemin d'un répertoire et affiche le contenu détaillé de celui-ci en faisant appel à la commande **ls** puis recommence à moins que l'utilisateur saisisse 0 pour mettre fin à l'exécution.

Exercice 13.4. Le menu (Examen 2024)

Vous devez réaliser un menu pour que l'utilisateur puisse choisir ce qu'il souhaite effectuer :

```
=== Menu ===

1. Afficher le contenu du répertoire courant
```

2. Créer un nouveau fichier vide

3. Quitter

Réalisez un programme en langage C pour UNIX/Linux affichant ce menu, réalisant l'action souhaitée par l'utilisateur puis réaffiche le menu jusqu'à ce que l'utilisateur quitte le programme.

Rappel : la commande **ls** permet de lister le contenu d'un répertoire et la commande **touch** permet de créer un fichier vide dont le nom est passé en paramètre.

Exemple d'exécution :

```
=== Menu ===
1. Afficher le contenu du répertoire courant
2. Créer un nouveau fichier vide
3. Quitter
1
menu  menu.c  menu.o
=== Menu ===
1. Afficher le contenu du répertoire courant
2. Créer un nouveau fichier vide
3. Quitter
2
Nom du fichier à créer ?
toto
=== Menu ===
1. Afficher le contenu du répertoire courant
2. Créer un nouveau fichier vide
3. Quitter
1
menu  menu.c  menu.o  toto
=== Menu ===
1. Afficher le contenu du répertoire courant
2. Créer un nouveau fichier vide
3. Quitter
3
```

14. Les signaux

Exercice 14.1. Questions express

Écrire une fonction qui pose une question à l'utilisateur qui doit répondre dans un temps impartit.

Écrire un programme posant 3 questions avec limite de temps.

Exercice 14.2. Le chronomètre (Examen 2024)

Vous devez réaliser un chronomètre (précis à une seconde près). Le lancement du programme démarre le chronomètre. L'utilisateur peut mettre en pause le chronométrage en appuyant sur *CTRL-Z*. Il utilise la même combinaison de touche pour reprendre le chronométrage. Il est également possible de relancer le chronométrage à partir de zéro en appuyant sur *CTRL-C*. Le programme se termine lorsque l'utilisateur appuie sur la touche *q* puis sur *Entrer*. Le temps du chronomètre n'est affiché que lorsque l'utilisateur met en pause le chronomètre, le relance ou le réinitialise.

Réalisez un programme en langage C pour UNIX/Linux pour cela.

Exemple :

```
^Z
4 -> en pause
^Z
4 -> c'est reparti…
^Z
7 -> en pause
^Z
7 -> c'est reparti…
^C
13 -> à la fin
0 -> c'est reparti…
^Z
3 -> en pause
q
```

15. La communication avec la mémoire partagée

Exercice 15.1. Le thermomètre

Toutes les secondes, un processus renseigne la température actuelle dans une zone de mémoire partagée. Comme nous n'avons pas accès à un véritable capteur, les valeurs sont en réalité des valeurs pseudo aléatoires. Au terme de 100 valeurs, une valeur particulière est positionnée dans la mémoire partagée afin d'indiquer que le capteur a terminé ses mesures.

Un autre processus lit dans la mémoire partagée la température toutes les 5 secondes. Il l'affiche sur la console.

Exercice 15.2. Calcul matriciel

Réaliser un programme permettant de réaliser une addition entre deux matrices. Le calcul de chaque valeur de la matrice résultat est effectué par un processus différent.

16. Les sémaphores

Exercice 16.1. Je sais compter !

Un enfant apprend à compter jusqu'à 100. Pour y aller progressivement son père l'aide en disant un chiffre sur deux. Le fils commence par dire 1 puis il attend que son père dise son nombre. Le père, de même attend que son fils ait indiqué son nombre pour donner le nombre suivant.

Réaliser un programme avec deux processus affichant de la même manière les nombres jusqu'à 100.

17. Les threads

Exercice 17.1. La somme des carrés entre 1 et N

Plutôt que calculer séquentiellement la somme des carrés entre 1 et N, appliquer la technique diviser pour régner : le travail est réparti entre 8 threads chacun s'occupant de calculer la somme des carrées d'un sous intervalle.

Exemple pour N = 1 000 000 :

```
Intervalle du thread :  500000001 à  625000000 ->          3.97135e+25
Intervalle du thread :  125000001 à  250000000 ->          4.55729e+24
Intervalle du thread :          1 à  125000000 ->          6.51042e+23
Intervalle du thread :  625000001 à  750000000 ->          5.92448e+25
Intervalle du thread :  750000001 à  875000000 ->          8.26823e+25
Intervalle du thread :  875000001 à 1000000000 ->          1.10026e+26
Intervalle du thread :  375000001 à  500000000 ->          2.40885e+25
Intervalle du thread :  250000001 à  375000000 ->          1.23698e+25
Somme des carrés entre          1 et 1000000000 :          3.33333e+26
```

Exercice 17.2. Le nombre de personne dans la pièce

Pour respecter la capacité maximale d'une pièce. Le nombre de personnes entrant et sortant de celle-ci est compté à chaque porte. Chaque porte est simulée à l'aide d'un thread. Aléatoirement simuler des entrées et des sorties à chaque porte en faisant attention à bien comptabiliser le nombre de personnes dans la pièce.

Exemple d'exécution :

```
Entrée de 1 personne par la porte A -> total : 1
Entrée de 2 personnes par la porte B -> total : 3
Entrée de 1 personne par la porte C -> total : 4
Entrée de 3 personnes par la porte D -> total : 7
Entrée de 1 personne par la porte B -> total : 8
Sortie de 4 personnes par la porte A -> total : 4
Entrée de 5 personnes par la porte B -> total : 9
Entrée de 1 personne par la porte D -> total : 10
1 personne bloquée à la porte C
Sortie de 6 personnes par la porte A -> total : 4
Entrée de 3 personnes par la porte B -> total : 7
Sortie de 6 personnes par la porte D -> total : 1
Sortie de 1 personne par la porte C -> total : 0
Entrée de 5 personnes par la porte D -> total : 5
Entrée de 1 personne par la porte C -> total : 6
Entrée de 2 personnes par la porte B -> total : 8
Sortie de 1 personne par la porte B -> total : 7
Sortie de 3 personnes par la porte C -> total : 4
Entrée de 1 personne par la porte A -> total : 5
Entrée de 2 personnes par la porte C -> total : 7
Sortie de 5 personnes par la porte D -> total : 2
Entrée de 1 personne par la porte B -> total : 3
Entrée de 2 personnes par la porte B -> total : 5
Entrée de 4 personnes par la porte A -> total : 9
4 personnes bloquées à la porte D
Sortie de 1 personne par la porte D -> total : 8
Sortie de 4 personnes par la porte D -> total : 4
Entrée de 3 personnes par la porte C -> total : 7
Sortie de 2 personnes par la porte C -> total : 5
Sortie de 4 personnes par la porte B -> total : 1
Sortie de 1 personne par la porte B -> total : 0
Entrée de 5 personnes par la porte D -> total : 5
Entrée de 5 personnes par la porte A -> total : 10
Sortie de 5 personnes par la porte C -> total : 5
Sortie de 1 personne par la porte A -> total : 4
Sortie de 2 personnes par la porte A -> total : 2
```