Représentation mémoire, modèle mémoire

I - Application du cours

I.1 - Booléens

Exercice 1

On se donne a , b et c trois booléens.

1. Indiquez le parenthésage de :
   a. NON a OU b
   b. a ET NON b OU c
   c. a ET b OU c ET a
   d. a OU b ET c OU a
2. Si a vaut Vrai, b vaut Faux, et c vaut Faux, que vaut :
   a. b ET c OU a
   b. NON(NON a ET (b OU NON c ET a))
   c. a ET b OU c ET a
   d. a OU b ET c OU a

Exercice 2

Simplifier le code C ci-dessous :

condition.c

```c
unsigned x = 0;
... // du code qui modifie x
if (!((x >= 296) && (x <= 305)) || ((x >= 313) && (x <= 338)) || ((x >= 345) && (x <= 370))) {
    ...
}
```

I.2 - Entiers

Exercice 3

On représente des entiers non-signés sur 2 octets. Complétez le tableau :

Binaire non-signé	Décimal	Hexadécimal
	42312	
	17213	
00110001 11110101		
11101010 00101001		
		48af
		1d85

Exercice 4

Les Shadoks vivent sur la planète Shadok[i]. Ils n'ont que quatre mots dans leur vocabulaire : GA, BU, ZO, MEU. Ils encodent donc le chiffre 0 par GA, 1 par MEU, 2 par ZO, 3 par MEU, et raisonnent en représentation non-signée en base 4 sur 4 chiffres (ils n'ont que quatre cases dans leur tête !).

1. Quel est le plus petit entier qu'un Shadok puisse exprimer ? Le plus grand ?
2. Exprimer $^{10}\overline{0}$, $^{10}\overline{4}$, $^{10}\overline{10}$ et $^{10}\overline{42}$ en shadok.
3. (Difficile[ii] !!)
   a. Démontrer que la notion de passoire est indépendante de la notion de trou, et inversement.
   b. Démontrer qu'un oeuf se compose uniquement et essentiellement d'extérieur.

i. Pour des raisons de coût des impressions, je me vois obligé d'abréger cette initiation rituelle.
ii. Ce sont des références à des épisodes de la série, pas de vraies question.

Exercice 5

Dans le jeu Fire Emblem Three Houses sorti en juillet 2019, les points de vie (pv) des personnages sont stockés dans un `uint8_t` . L'un des boss finaux posséde 199 pv, et se régénère de 60 pv au début de son tour. Sa régénération est codée à peu près de cette façon :

FE3H.c
```c
uint8_t nouveauPV = min(ancienPV+60, 199);
```

Le jeu fonctionne au tour par tour. À leur tour, les joueurs diminuent les pv du boss. Ensuite c'est au tour du boss, qui commence par se régénérer, puis diminue les pvs des joueurs (aïe). Puis c'ets à nouveau aux joueurs, etc.

1. En admettant qu'il est simple d'enlever peu de pv et difficiles d'enlever beaucoup de pv au boss, proposez une stratégie permettent de tuer le boss (c'est à dire de faire en sorte qu'il ait 0 pv ou moins) très voire trop facilement.
2. En février 2020, cette formule de régénération a été changée pour prévenir cette stratégie. Proposez une telle modification de la formule.

I.3 - Nombres à virgule flottante

Exercice 6

Que va-t-il se passer si l'on exécute ce code ? Pourquoi ?

exp.c
```c
/** Recherche un zéro de la fonction exponentielle dans les négatifs
 */
int main() {
    double x = 0;
    while (exp(x) > 0) {
        x = x-1;
    }
    printf("%f", x);

    return 0;
}
```

NB : la fonction `double exp(double x)` calcule e^x .

Exercice 7

1. Écrire en base 10 les nombres (binaires) suivant : 0,101 ; 101,01 ; 110,011.
2. Écrire en base 2 les nombres (décimaux) suivants : 0,125 ; 2,5 ; 13,875.
3. En déduire les notations mantisse - exposant de : 0,125 ; 2,5 et 13,875, en base 10 et en base 2.

Remarque : dans cet exercice, tous les nombres donnés sont des nombres dyadiques, ils s'écrivent en base 2 de manière finie. Mais ce n'est pas le cas de tous les décimaux, et encore moins de tous les nombres.

4. Écrire le nombre décimal 0,1 en binaire avec 4 chiffres après la virgule.

Exercice 8

On veut écrire 13,875 en représentation machine en simple précision[i]. Le principe est le même que pour la double précision, mais sur 32 bits au total au lieu de 64. La répartition est la suivante :
1 bit de signe, 8 bits pour l'exposant avec une constante d'excentrement de 127 ($2^7 - 1$) et 23 bits pour la mantisse.

1. Écrire l'exposant de 13,875 en base 2, puis le décaler de 127.
2. En déduire la représentation machine de 13,875 en simple précision.

i. Pas parce que c'est plus intéressant, parce que c'est plus rapide à écrire à la main.

Exercice 9 – *C'est vraiment de l'info ??*

Un MP2I veut peser ses cahier de maths pour découvrir combien de savoir il a assimilé. Il utilise pour cela une balance à trois chiffres significatifs (en base 10). Il se doute que l'ordre de grandeur de la masse des cours est celui du kilo.
Cependant, *le vent se lève*[i] et les feuilles ne tiennent pas sur la balance. L'élève se pèse donc lui-même en tenant les feuilles, puis lui-même sans les feuilles. Avec quel précision obtient-il la masse des cours ?

i. *Il faut tenter de peser.*

I.4 - Modèle mémoire

Exercice 10

On considère l'exécution du programme C ci-dessous :

indice-maxi.c

```c
bool str_inf(int a, int b) {
  return a < b;
}

int indice_maxi_tableau(int T[], int len) {
  if (len == 0) { exit(EXIT_FAILURE); }

  int best_indice = 0;
  int indice = 1;
  while (indice < len) {
    if ( str_inf(T[best_indice], T[indice]) ) {
      best_indice = indice;
    }
    indice = indice +1;
  }

  return best_indice;
}

int main(void) {

  int T[] = {5, -1, 55, -2, 34, 65, 12};
  int indice_maxi = indice_maxi_tableau(T, 5);
  printf("La plus grande des %d premières valeurs est : %d\n",
          5, indice_maxi);

  return EXIT_SUCCESS;
}
```

1. Dans quelle zone mémoire est stockée la chaine de caractère passée en argument à `printf` ?
2. Représenter l'état de la mémoire lorsque :
   a. Juste avant le début de la ligne 27.
   b. Fin de la ligne 15.

   c. Première fois que l'on atteint le début de la ligne 7.
   d. Première fois que l'on atteint le début de la ligne 20.
   e. Durant l'appel de `printf` ligne 31.
   f. Juste après la fin de la ligne 34.
3. On ne veut plus utiliser des tableaux mais des zones allouées avec `malloc`.
   a. Indiquer ce qu'il faut changer.
   b. Refaire les sous-qestions précédentes dont la réponse diffère désormais.

II - Plus avancé

Exercice 11 – *Distributivité des opérateurs logique*

Une **table de vérité** est le tableau des valeurs que prend une formule logique en fonction de la valeur de ses variables.
On se donne a , b et c trois booléens. En écrivant les tables de vérité, montrez que :

1. Distributivité du ET sur le OU, et récpiroquement :
   a. a ET (b OU c) a la même valeur de vérité que (a ET b) OU (a ET c) .
   b. a OU (b ET c) a la même valeur de vérité que (a OU b) ET (a OU c) .
   c. Et si l'on s'intéresse à (b OU c) ET a ? Idem pour (b ET c) OU a .
2. Lois de Morgan (distributivité du NON sur ET et OU) :
   a. NON(a ET b) a la même valeur de vérité que NON(a) OU NON(b) .
   b. NON(a OU b) a la même valeur de vérité que NON(a) ET NON(b) .

Exercice 12 – *Addition et soustraction en complément à 2*

On s'intéresse ici aux algorithmes d'addition et de soustraction en complément à 2.
Pour simplifier, on travaille en base 10 et en complément à 10. On supposera les nombres écrits sur 3 "bits" décimaux. Ainsi, le code "000" représente $^{10}\overline{0}$, le code "001" représente $^{10}\overline{1}$, etc, le code "499" représente $^{10}\overline{499}$, le code "500" représente $^{10}\overline{-500}$ (remarquez que 500 = 1000-500), le code "501" représente $^{10}\overline{-499}$ (remarquez que 501 = 1000-499), etc, le codde "999" représente $^{10}\overline{-1}$ (remarquez que 999 = 1000-1) .
Soient "abc" et "xyz" deux codes.

1. Quels sont les plus petits et plus grands entiers représentables ? Comment s'écrit $^{10}\overline{96}$ et $^{10}\overline{-103}$? Et $^{10}\overline{17}$? $^{10}\overline{-2}$? Et $^{10}\overline{456}$?
2. Pour calculer l'addition, on additionne les codes des entiers (avec l'algorithme appris à l'école primaire) et on ignore les retenues sortantes.
   a. Calculer $^{10}\overline{17} + {}^{10}\overline{96}$.
   b. Caculer $^{10}\overline{96} + {}^{10}\overline{(-103)}$.
   c. Calculer $^{10}\overline{17} + {}^{10}\overline{(-2)}$.
   d. Calculer $^{10}\overline{-103} + {}^{10}\overline{(-2)}$.
   e. Calculer $^{10}\overline{96} + {}^{10}\overline{456}$.

Pour la soustraction, c'est plus subtil et plus magique. Pour calculer abc-xyz on calcule abc+(1000-xyz) : comme on ignore les retenues sortantes, dans le résultat final le 1 de 1000 disparaitra.

3. La soustraction des codes "999" - "abc" est simple : peut-elle créer une retenue ? Comment la caculer facilement ?
4. Donner alors un algorithme pour calculer abc-xyz.

Exercice 13 – *Détection de débordements*

Trouvez une condition nécessaire et suffisante permettant de détecter des dépassements de capacité dans l'algorithme d'addition en complément à 2.
Indice : Vous pouvez vous intéresser au signe des opérandes et du résultat de l'algorithme d'addition.

Exercice 14

Soit $x \in \mathbb{R}$.

1. Prouver que si x est dyadique, alors il est décimal.
2. Prouver que la réciproque est fausse en exhibant un contre-exemple.

Exercice 15 – *Entiers signés par bits de signe*

Dans cet exercice, on encode les entiers sur 8 bits naïvement en utilisant un bit de signe. Le bit le plus à gauche représente le signe (0 pour -, 1 pour +) et les 7 bits restants encodent un entier positif en base 2.

1. Écrire 6 et -11 en base 2 dans cette représentation.
2. Calculer 6 + (-11) en faisant naïvement l'addition bit à bit (comme si les encodages représentaient des entiers non signés). Qu'obtenez-vous ?
3. Mêmes questions en complément à 2 (sur 8 bits toujours). Que remarquez-vous ?
4. Rappelez l'algorithme de CE1-CE2 pour la soustraction.

Point culture générale : si l'addition s'implémente bien dans un processeur, la soustraction est plus compliquée - et certainement pas aussi agréable qu'en complément à 2.

Exercice 16 – *Complément à 1*

Le complément à 1 est un troisième encodage possible des entiers relatifs en binaire :

- $n \geq 0$ est encodé par son écriture en base 2
- $n \leq 0$ est encodé par le complémentaire bit à bit de son écriture en 0.

Ainsi, sur 4 bits le nombre $^{10}\overline{1}$ s'écrit "0001" et $^{10}\overline{1110}$. Pour les questions suivantes, on supposera que l'on réalise cet encodage sur 4 bits, et que l'on encode autant d'entiers strictement positifs que d'entiers strictement négatifs.

1. Quel est le plus grand entier représentable ? Le plus petit ?
2. A-t-on un moyen simple de savoir si un entier est positif ou s'il est négatif ?
3. Existe-t-il un code qui représente plusieurs entiers ? Ou un entier qui est représenté par plusieurs codes ?
4. L'addition des codes donne-t-elle le code de l'addition ? Et pour la soustraction ?

Point culture générale : si l'addition s'implémente bien dans un processeur, la soustraction est plus compliquée - et certainement pas aussi agréable qu'en complément à 2.