

IFT209 – Programmation système
Université de Sherbrooke

Devoir 1

Enseignant: Michael Blondin
Date de remise: mercredi 18 janvier 2023 à 23h59
À réaliser: en équipe de deux
Modalités: remettre en ligne sur **Turnin** dans un fichier PDF
Bonus: les questions bonus sont indiquées par ★
Pointage: max. 30 points + 1,5 points bonus

Question 1: changements de base

9 pts

Effectuez les conversions ci-dessous avec l'approche *la plus directe* parmi celles présentées en classe (et dans les notes de cours), c'est-à-dire: $b \rightarrow 10$, $10 \rightarrow b$, $b^m \rightarrow b$, $b \rightarrow b^m$, $b^m \rightarrow b^k$ ou $b \rightarrow b'$. Donnez vos résultats sans zéros non significatifs. Laissez une trace de votre démarche.

- (a) 314256 de la base 7 vers la base 10
- (b) 695581 de la base 10 vers la base 16
- (c) 34152 de la base 6 vers la base 3
- (d) 100111110001101 de la base 2 vers la base 16
- (e) 5FB1C7 de la base 16 vers la base 8
- (f) 2637 de la base 8 vers la base 2

Question 2: compréhension des systèmes de numération

6 pts

- (a) Quel est le plus petit nombre de bits nécessaire afin de représenter $38E457F99BAF_{16}$ en binaire? Justifiez.
- (b) Soit $n \geq 2$ et soit x le plus grand nombre représentable avec n chiffres dans le système octal. Expliquez pourquoi x_{16} termine forcément par le chiffre F.
- (c) Écrivez le nombre fractionnaire décimal $147/8$ sous sa forme binaire. Expliquez comment vous avez obtenu votre réponse.

Question 3: addition

4 pts

Effectuez ces additions directement dans la base indiquée, c.-à-d. *sans* convertir dans une base intermédiaire. Laissez une trace de vos calculs.

(a)
$$\begin{array}{r} 110110_2 \\ + 10110101_2 \\ \hline \end{array}$$

(b)
$$\begin{array}{r} 7A2D56_{16} \\ + B99C_{16} \\ \hline \end{array}$$

Question 4: système unaire et initiation à la programmation de bas niveau

Le but de cette question est d'implémenter deux opérations dans le système unaire: le *carré* et le *maximum*. Vous devez donner vos procédures sous forme de programmes dans le langage d'assemblage artificiel UNARISC.

Le langage UNARISC offre cinq registres: r_1 , r_2 , r_3 , r_4 et r_5 . Chaque registre peut contenir la séquence vide ou une séquence de taille arbitraire formée uniquement du symbole 1. Autrement dit, chaque registre représente un nombre en notation unaire. Un programme UNARISC est constitué d'étiquettes et de ces instructions:

Instruction		Effet
push	r	ajoute un symbole au registre r
pop	r	retire un symbole du registre r s'il n'est pas vide, sinon ne fait rien
empty	r, l	saute à l'étiquette l si le registre r est vide, sinon ne fait rien
goto	l	saute à l'étiquette l
concat	r, s	ajoute autant de symboles au registre r que le registre s en contient
ret	r	termine l'exécution du programme en retournant le contenu du registre r

Par exemple, le programme suivant calcule l'addition des registres r_1 et r_2 :

```

début:  empty  r1, fin
        pop    r1
        push   r2
        goto   début
fin:     ret    r2

```

- (a) Donnez un programme UNARISC qui retourne $(r_1)^2$ en notation unaire. Par exemple, si initialement $r_1 = 111$, alors le programme doit retourner 111111111. Supposez que les registres r_2 , r_3 , r_4 et r_5 sont initialement vides. Selon la complexité de votre code, commentez-le pour qu'il soit compréhensible. 3 pts
- (b) Donnez un programme UNARISC qui retourne $\max(r_1, r_2)$ en notation unaire. Par exemple, si initialement $r_1 = 11111$ et $r_2 = 111$, alors le programme doit retourner 11111. Supposez que les registres r_3 , r_4 et r_5 sont initialement vides. Selon la complexité de votre code, commentez-le pour qu'il soit compréhensible. 3 pts

Remarque: si vous n'y arrivez pas, implémentez plutôt $\min(r_1, r_2)$ pour obtenir la moitié des points.

- ★ Donnez un programme UNARISC qui retourne 2^{r_1} en notation unaire. Par exemple, si initialement $r_1 = 111$, alors le programme doit retourner 111111111. Supposez que les registres r_2 et r_3 sont initialement vides. Vous ne devez utiliser ni l'instruction **concat**, ni les registres r_4 et r_5 . ★ 1,5 pts

Question 5: architecture des ordinateurs

Considérons une architecture qui utilise le format « little-endian » (petit-boutiste). Supposons que la mémoire principale contienne ces données:

adresse	contenu
⋮	⋮
5F17 ₁₆	13 ₁₆
5F18 ₁₆	3B ₁₆
5F19 ₁₆	F8 ₁₆
5F1A ₁₆	70 ₁₆
5F1B ₁₆	C8 ₁₆
5F1C ₁₆	41 ₁₆
5F1D ₁₆	5D ₁₆
⋮	⋮

(a) Quelle est la valeur *décimale* du mot stocké à l'adresse 5F1A₁₆? Justifiez votre réponse.

2,5 pts

(b) Le mot stocké à l'adresse 5F1A₁₆ est-il à une adresse *alignée*? Justifiez votre réponse.

2,5 pts