

IFT209 – Programmation système

Université de Sherbrooke

Devoir 4

Enseignant: Michael Blondin
 Date de remise: dimanche 3 avril 2022 à 23h59
 À réaliser: en équipe de deux
 Modalités: remettre en ligne sur **Turnin**
 Pointage: sur 20 points + 1,5 point bonus

Ce devoir cherche à mettre en pratique la manipulation de chaînes de bits et de caractères, ainsi que la mise au point de sous-programmes.

Problème. Afin de mieux comprendre le fonctionnement interne des bibliothèques de chaînes de caractères, cherchons à implémenter diverses manipulations de chaînes de caractères. Lorsque votre programme est exécuté, il doit lire une chaîne de caractères s au clavier, puis un entier n compris entre 0 et 5 (inclusivement). L'opération n est ensuite effectuée sur s , puis le programme se termine. Vous devez implémenter ces six opérations:

Taille d'une chaîne	
Opération	0
Entrée	Chaîne de caractères s sous codage UTF-8
Effet	Affiche le nombre de caractères (non nuls) de s
Tests	abcdef → 6
	é → 1
	aéケ▷ → 4

Remarquez que cette opération ne requiert des connaissances que sur les manipulations de bits. En effet, une chaîne de caractères UTF-8 s est un tableau d'octets dont le dernier octet est nul. La taille peut être calculée à l'aide de cet algorithme:

```

taille = 0

faire:
  charger l'octet actuel c du tableau s

  si c = 00000000:      retourner taille
  sinon si c débute par 0:  k = 1
  sinon si c débute par 110: k = 2
  sinon si c débute par 1110: k = 3
  sinon si c débute par 11110: k = 4

  taille += 1
  avancer de k octets dans le tableau s
  
```

Casses et substitutions	
Opération	1
Entrée	Chaîne de caractères s sous codage ASCII
Effet	Affiche la chaîne obtenue en appliquant ces opérations à s: - Les lettres aux positions paires sont mises en minuscule et les lettres aux positions impaires sont mises en majuscule (la première position est 0); - Ces voyelles doivent être remplacées par ces chiffres: A ↦ 4 E ↦ 3 I ↦ 1 O ↦ 0 a ↦ 4 e ↦ 3 i ↦ 1 o ↦ 0 - Les autres caractères sont inchangés.
Tests	brUN → bRuN bonjour! → b0nJ0Ur! Ceci est une phrase formidable → c3c1 3sT Un3 PhR4S3 f0rM1D4Bl3

Hexadécimal vers décimal	
Opération	2
Entrée	Chaîne de caractères s sous codage ASCII représentant un nombre hexadécimal non signé préfixé par « 0x » (valeur comprise entre 0 et $2^{64} - 1$ inclusivement)
Effet	Affiche la valeur décimale de s
Tests	0x5 → 5 0x0A → 10 0xFF → 255 0xABCDEF98 → 2882400152

Binaire vers décimal	
Opération	3
Entrée	Chaîne de caractères s sous codage ASCII représentant un entier signé binaire préfixé par « 0b » (valeur comprise entre -2^{63} et $2^{63} - 1$ inclusivement)
Effet	Affiche la valeur décimale de s
Tests	0b0101 → 5 0b00111 → 7 0b10100 → -12 0b111111 → -1

Chiffrement par décalage																
Opération	4															
Entrée	Chaîne de caractères s sous codage ASCII dont les caractères sont parmi: A, B, C, D, E, F, H, I, J, K, L, M, N, P, Q, R, S, T, U, V, X, Y, Z, [, \ et]															
Effet	La chaîne s a été chiffrée à partir d'une chaîne t. Vous devez afficher la chaîne t; autrement dit, vous devez <i>déchiffrer</i> s. La chaîne t est constituée exclusivement de lettres majuscules (A à Z). La chaîne s a été obtenue à partir de t en appliquant les deux transformations consécutives suivantes à chacune des lettres de t: (i) La lettre est décalée circulairement de 7 positions vers l'avant dans l'alphabet: <table><tr><td>A</td><td>↦</td><td>H</td></tr><tr><td>B</td><td>↦</td><td>I</td></tr><tr><td>⋮</td><td>↦</td><td>⋮</td></tr><tr><td>Y</td><td>↦</td><td>F</td></tr><tr><td>Z</td><td>↦</td><td>G</td></tr></table> (ii) Les 5 bits de poids faible du caractère obtenu à l'étape (i) sont décalés circulairement de 3 bits vers la gauche. Exemple de la transformation: (i) C ↦ J (ii) J = 01001010₂ ↦ 01010010₂ = R <i>Attention:</i> vous devez effectuer la procédure <i>inverse</i> afin de retrouver t	A	↦	H	B	↦	I	⋮	↦	⋮	Y	↦	F	Z	↦	G
A	↦	H														
B	↦	I														
⋮	↦	⋮														
Y	↦	F														
Z	↦	G														
Tests	<table><tr><td>R</td><td>→</td><td>C</td></tr><tr><td>RRR</td><td>→</td><td>CCC</td></tr><tr><td>HU]VCRNCH</td><td>→</td><td>TOPSECRET</td></tr></table>	R	→	C	RRR	→	CCC	HU]VCRNCH	→	TOPSECRET						
R	→	C														
RRR	→	CCC														
HU]VCRNCH	→	TOPSECRET														

Permutations	
Opération	5
Entrée	Chaîne de caractères s sous codage ASCII sans répétition de caractères
Effet	Affiche toutes les permutations de s (dans l'ordre de votre choix) <i>Indice: pensez à une procédure récursive.</i>
Tests	ab → ab ba abc → abc acb bac bca cba cab

Directives.

- Votre programme doit être obtenu en complétant le code partiel ci-dessous;
- Votre programme doit être remis dans un seul fichier nommé `devoir4.s`;
- Ne modifiez pas le point d'entrée ainsi que le format de la chaîne en entrée;
- Supposez que la chaîne *s* respecte le format de l'opération choisie (aucune validation);
- N'utilisez *pas* `printf`, `scanf` ou d'autres fonctions d'une librairie afin d'implémenter les opérations.

Pointage. Vous pouvez obtenir jusqu'à 20 points répartis ainsi:

- 1 point pour la lecture d'une chaîne de caractères et d'un code d'opération, et l'affichage d'un résultat;
- 1 point pour chaque opération qui passe les tests donnés plus haut (donc 6 points au maximum);
- 1,5 points par opération bien implémentée (donc 9 points au maximum);
- 2 points pour l'indentation du code (codes d'opération, opérandes et commentaires alignés);
- 2 points pour la qualité et lisibilité du code (commentaires significatifs, usage des registres, organisation du code, pas de « code spaghetti », etc.)

Bonus. Vous obtenez 0,75 point bonus pour chacune de ces fonctionnalités additionnelles:

- l'opération 3 est implémentée sous forme de *sous-programme* avec *au plus dix* instructions (excluant étiquettes/commentaires), *aucune* instruction arithmétique, *au plus une* occurrence de **cbz**, *au plus une* occurrence de **b**, et *aucune* autre instruction de branchement (**b.cond**, **bl**, **blr**, **cbnz**, **tbz**, **tbnz**, etc.)
- l'opération 5 supporte les caractères UTF-8 (par exemple: `aéケ` → `aéケ aケé éaケ éケa ケéa ケaé`)

Code partiel.

```
.include "macros.s"
.global main

main:
    adr    x0, fmtLecture
    adr    x1, chaine
    bl     scanf

    mov    x0, 0
    bl     exit

.section ".data"
// Mémoire allouée pour une chaîne de caractères d'au plus 1024 octets
chaine:    .skip    1024

.section ".rodata"
// Format pour lire une chaîne de caractères d'une ligne (incluant des espaces)
fmtLecture: .asciz    "%[^\n]s"
```