Nom :	Prénom :
N° Étudiant :	Groupe de TD :

TME Solo 2023 – 2024 – Sujet n°2

Architecture des ordinateurs 1 – LU3IN029

Durée : 0h55

Documents autorisés : Aucun document ni machine électronique n'est autorisé à l'exception du mémento MIPS.

Le barème indiqué pour chaque question n'est donné qu'à titre indicatif. Le barème total est lui aussi donné à titre indicatif. Merci de rendre la feuille.

Étapes préliminaires et consignes à suivre scrupuleusement :

1. Créez un répertoire pour le TME solo à la racine de votre compte qui devra contenir les codes réalisés, en tapant une à une et dans l'ordre les commandes suivantes (ce qui est après le signe > ci-dessous) dans un terminal :
   `> cd`
   `> mkdir TMESolo_<nom>` (`<nom>` est à remplacer par votre nom en minuscule sans espace ni accent)
   `> cd TMESolo_<nom>` (`<nom>` est à remplacer par votre nom en minuscule sans espace ni accent)
   `> chmod -R go-rwx .` (copier strictement cette commande, le "." inclus)
   Attention : cette dernière commande est très importante car elle empêche d'autres utilisateurs d'accéder à vos fichiers. Si vous ne la faites pas correctement et qu'un autre étudiant copie vos fichiers, vous risquez d'obtenir la note de 0.
   Remarque : la détection de plagiat sera faite automatiquement par logiciel.
2. **Important :** indiquez en commentaire dans tous vos fichiers assembleur vos nom, prénom et numéro d'étudiant.
3. Lancez Mars (commande `mars` ou commande `java -jar /usr/local/mars/Mars4_5.jar`) et composez le TME solo en répondant aux questions ci-dessous. Enregistrez bien tous vos codes dans le répertoire `TMESolo_<nom>`.

Soumission de votre devoir à la fin du TME solo

1. Créez une archive contenant vos codes réponses avec les commandes suivantes :
   `> cd`
   `> tar -cvf tmesolo_<nom>.tar TMESolo_<nom>/`
2. Déposez l'archive dans Moodle : dans la section "TME solo : information et organisation" il y a une remise de devoir intitulée "Remise TME solo de 9h35...". Deux tentatives sont autorisées au cas où vous vous tromperiez de fichier. Attention vous devez soumettre avant 10h35, si vous modifiez votre rendu après cette heure vous serez en retard et pénalisé.

Le TME solo est composé d'un exercice sur 21 points : la première question est sur 8 points, la deuxième sur 5 et la troisième sur 8. Vous devez répondre aux questions dans l'ordre.

Exercice 1 : Construction d'un programme assembleur MIPS construisant une chaîne de caractères en miroir d'une chaine initiale – 21 points

On considère le programme C donné ci-dessous. Ce programme déclare une variable globale initialisée de type chaîne de caractères, `str`, en calcule la longueur par appel à la fonction `len`, puis construit la chaîne `str_res` dans laquelle l'ordre des lettres est inversé, par appel à la fonction `miroir`, et enfin affiche la chaine nouvellement construite.

Par exemple, l'exécution du programme ci-dessous produit l'affichage "edcba".

```
char str[] =  "abcde";
char str_res[20];

int len(char * str) {

  // ... voir question 2
}


void miroir(char * str_src, char * str_dst, int idx_src) {

  // ... voir question 3
}


int main() {
    int n;

    n = len(str);

    miroir(str, str_res, n - 1);

    printf("%s", str_res);

    return 0;
}
```

On se propose d'écrire le programme assembleur MIPS associé, en différentes étapes.

Consignes importantes :

- Il vous est demandé de mettre des commentaires dans vos codes pour indiquer la correspondance entre les registres utilisés et les variables des programmes.
- Les variables locales peuvent être optimisées en registre et globalement votre code peut être optimisé **mais** vous devez suivre scrupuleusement les conventions habituelles d'utilisation des registres et du cours. Il n'est pas demandé de sauvegarder les registres persistants, ni `$31`, dans le `main`.
- Toute allocation en pile devra être assortie d'un commentaire justifiant le nombre d'octets alloués.

Question 1.1 : 8 points

Dans un fichier nommé **`Q1.s`** (et enregistré dans le répertoire pour le TME solo), écrivez un programme assembleur correspondant à la section `data`, et à la section `text` comprenant le programme principal (`main`) uniquement. Lors des appels à `len` et `est_palin`, `str` désigne l'adresse du premier élément de la chaine `str`.

Testez votre programme pour différentes valeurs de la chaine `str`. Pour cela, il est nécessaire d'implanter deux fonctions minimales pour `len` et `miroir` (uniquement l'instruction de retour de fonction). Décrivez le contenu du segment de données, octet par octet, pour la zone correspondant à `str`.

La fonction `len` est définie comme suit. La chaine est parcourue de son premier à son dernier élément, et un compteur est incrémenté à chaque nouvel élément rencontré. A l'issue du parcours, la valeur du compteur est renvoyée à l'appelant.

```
int len(char * str) {
    int c = 0;
    while (str[c] != '\0') {
        c += 1;
    }
    return c;
}
```

Question 1.2 : 5 points

Enrichissez votre programme en complétant la fonction `len`. Vous sauvegarderez cette version du programme dans le fichier **`Q2.s`**

Testez votre programme pour vérifier qu'il fonctionne. Vous décrirez les tests réalisés.

La fonction `miroir` est définie comme suit. La chaine initiale est parcourue de son dernier élément jusqu'à son premier élément. A chaque nouvel élément accédé, cet élément est recopié dans la chaîne résultat, qui elle est parcourue du premier au dernier élément.

```
void miroir(char * str_src, char * str_dst, int idx_src) {
    int idx_dst = 0;
    while (idx_src >= 0) {
        str_dst[idx_dst] = str_src[idx_src];
        idx_dst += 1;
        idx_src -= 1;
    }
}
```

Question 1.3 : 8 points

Enrichissez votre programme en complétant la fonction `miroir`. Vous sauvegarderez cette version du programme dans le fichier **`Q3.s`**

Testez votre programme pour vérifier qu'il fonctionne. Vous préciserez les tests effectués.