IFT209 – Programmation système

Université de Sherbrooke

Laboratoire 6

Enseignant:	Michael Blondin
Date de remise:	dimanche 31 mars 2019 à 23:59
À réaliser:	en équipe de deux
Modalités:	remettre en ligne sur Turnin; une seule remise avec vos noms/CIP en commentaires en en-tête du code

Problème. La *racine carrée* d'un nombre $x \in \mathbb{R}_{\geq 0}$ est l'unique nombre non négatif $r \in \mathbb{R}_{\geq 0}$ tel que $r^2 = x$. Nous écrivons $\sqrt{x}$ afin de dénoter r. Le but de ce laboratoire est d'écrire un programme qui calcule la racine carrée d'un nombre sans utiliser l'instruction `fsqrt` (ou toute autre instruction d'exponentiation).

Rappelons d'abord que la racine carrée d'un nombre peut être irrationnelle, par ex. $\sqrt{2}$. Ainsi, il est impossible de calculer précisément $\sqrt{x}$ en nombre en virgule flottante. Vous devez donc calculer un nombre y qui approxime $\sqrt{x}$. Au laboratoire 4, nous avons utilisé la recherche dichotomique afin de rechercher un élément dans un tableau trié en temps logarithmique. Cette approche, qui ne semble peut-être pas connexe, permet aussi d'approximer une racine carrée.

Nous supposerons pour le reste de l'énoncé que $x \geq 1$. Observons que:

$$1 \leq \sqrt{x} \leq x. \tag{1}$$

Tentons de calculer $\sqrt{6{,}25}$. Par l'identité (1), nous savons que la racine carrée se situe dans l'intervalle $[1; 6{,}25]$. Ainsi, comme première approximation, prenons la valeur qui se situe à mi-chemin, c'est-à-dire $y = 3{,}625$. Puisque $y^2 = 13{,}140625 > 6{,}25$, notre approximation est trop grande. Il faut donc réduire y. Ainsi, la racine carrée se situe dans l'intervalle $[1; 3{,}625]$. Comme nouvelle approximation, prenons la valeur qui se situe à mi-chemin du nouvel intervalle, c'est-à-dire $y = 2{,}3125$. Puisque $y^2 = 5{,}34765625 < 6{,}25$, notre approximation est trop petite. Il faut donc augmenter y. Ainsi, la racine carrée se situe dans l'intervalle $[2{,}3125; 3{,}625]$. En réduisant ainsi l'intervalle à répétition, nous nous approchons progressivement de la racine carrée qui est 2,5:

$$\begin{array}{ll}
[1 & ; 6{,}25] \\
[1 & ; 3{,}625] \\
[2{,}3125 & ; 3{,}625] \\
[2{,}3125 & ; 2{,}96875] \\
[2{,}3125 & ; 2{,}640625] \\
[2{,}4765625 & ; 2{,}640625] \\
[2{,}4765625 & ; 2{,}55859375] \\
[2{,}4765625 & ; 2{,}517578125] \\
\vdots & \vdots
\end{array}$$

Nous pouvons donc nous arrêter lorsque l'erreur absolue de l'approximation est d'au plus ϵ, pour une petite valeur ϵ de notre choix; autrement dit, lorsque l'intervalle $[a, b]$ obtenu est tel que $(b - a)/2 \leq \epsilon$.

À implémenter. Vous devez implémenter un programme qui lit deux nombres x et ϵ en virgule flottante double précision, et qui affiche une approximation de $\sqrt{x}$ à l'aide de l'approche décrite ci-dessus. Le calcul de $\sqrt{x}$ doit être implémenté sous forme de sous-programme avec x et ϵ en paramètres.

Tests. Vous pouvez tester votre programme en choisissant une petite valeur de ϵ, par ex. $\epsilon = 0{,}00005$, et en vérifiant que vous vous approchez bien à une distance d'au plus ϵ de ces valeurs:

$$\sqrt{1} = 1$$
$$\sqrt{4} = 2$$
$$\sqrt{9} = 3$$
$$\sqrt{6{,}25} = 2{,}5$$
$$\sqrt{1523064{,}51563} = 1234{,}125$$

Directives.

— Votre programme doit être obtenu en complétant le code partiel ci-bas;
— Votre programme doit être remis dans un seul fichier nommé `labo6.s`;
— Ne modifiez pas le point d'entrée ainsi que le format des entrées et sorties;
— Vous ne pouvez pas utiliser l'instruction `fsqrt` ou toute autre instruction d'exponentiation;
— Vous pouvez supposer que les valeurs en entrée sont valides, et en particulier que $x \geq 1$ et $\epsilon > 0$.

Pointage. Vous pouvez obtenir un maximum de 10 points. Vous obtenez:

— 1 point si votre programme lit deux nombres en virgule flottante;
— 1 point si votre programme affiche un nombre en virgule flottante;
— 5 points si votre programme approxime bien $\sqrt{x}$ à une erreur absolue d'au plus ϵ;
— 1,5 points si votre code est bien indenté (codes d'opération, opérandes et commentaires alignés);
— 1,5 points pour la qualité et lisibilité du code (commentaires significatifs, organisation du code, etc.)

Code partiel.

```
.global main

main:
        // Lire x

        // Lire ε

        // r = racine(x, ε)

        // Afficher r

        mov     x0, 0
        bl      exit

racine:
        /* récupérer x de d0 et ε de d1 */
        ret
```