IFT209 – Programmation système

Université de Sherbrooke

Laboratoire 5

Enseignant: Michael Blondin
Date de remise: *aucune*
Modalités: *aucune*

Le but de ce laboratoire est de mettre en pratique l'arithmétique en virgule flottante.

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. Cherchons à calculer la racine carrée d'un nombre sans utiliser l'instruction `fsqrt` (ou toute autre instruction d'exponentiation).

Puisque la racine carrée d'un nombre peut être irrationnelle, par ex. $\sqrt{2}$, il est impossible de la calculer précisément en nombre en virgule flottante. Vous devez donc calculer un nombre y qui *approxime* $\sqrt{x}$. Pour ce faire, nous utiliserons une *recherche dichotomique*.

Supposons pour le reste de l'énoncé que $x \geq 1$ et ainsi que $1 \leq \sqrt{x} \leq x$. Tentons de calculer $\sqrt{6{,}25}$. Nous savons qu'elle se situe dans l'intervalle $[1; 6{,}25]$. Comme première approximation, prenons la valeur qui se situe à mi-chemin, c.-à-d. $y = 3{,}625$. 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.-à-d. $y = 2{,}3125$. 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 continuant ainsi, nous nous approchons progressivement de la racine carrée qui est $2{,}5$:

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

Nous pouvons 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 $(b - a)/2 \leq \varepsilon$.

Implémentation. Vous devez implémenter un programme qui approxime $\sqrt{x}$ à l'aide de l'approche décrite:

Entrées : nombres en virgule flottante $x \geq 1$ et $\varepsilon \in (0, 1)$
Sorties : approximation de $\sqrt{x}$ à une erreur absolue d'au plus ε

$a \leftarrow 1$
$b \leftarrow x$
$c \leftarrow (a + b)/2$
tant que $(b - a)/2 > \varepsilon$ **faire**
  si $c^2 < x$ **alors** $a \leftarrow c$
  sinon $b \leftarrow c$
  $c \leftarrow (a + b)/2$
retourner c

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

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

Directives.

— Vous devez compléter le code partiel ci-dessous;
— 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;
— Supposez que les valeurs en entrée sont valides, et en particulier que $x \geq 1$ et $0 < \varepsilon < 1$.

Code partiel.

```
.global main

main:                                   // main()
    // Lire x                           // {
    adr     x0, fmtLecture              //
    adr     x1, temp                    //
    bl      scanf                       //   scanf("%lf", &temp)
    ldr     d8, temp                    //   x = temp
                                        //
    // Lire ε                           //
    adr     x0, fmtLecture              //
    adr     x1, temp                    //
    bl      scanf                       //   scanf("%lf", &temp)
    ldr     d9, temp                    //   ε = temp
                                        //
    // Calculer racine carrée de x      //
    /*  Code à compléter  */            //   r = racine(x, ε)
                                        //
    // Afficher racine carrée           //
    adr     x0, fmtSortie               //
    fmov    d0, d8                      //
    bl      printf                      //   printf("%lf\n", r)
                                        //
    // Quitter programme                //
    mov     x0, 0                       //
    bl      exit                        //   return 0
                                        // }
                                        //
racine:                                 // racine(double x, double ε)
    /* Récupérer x de d0 et ε de d1,    // {
       et retourner racine dans d0  */  // }

.section ".rodata"
fmtLecture: .asciz  "%lf"
fmtSortie:  .asciz  "%lf\n"

.section ".bss"
            .align  8
temp:       .skip   8
```