Algorithmique 6 : numération

Exercice 1 : conversion binaire vers décimal

On veut calculer la valeur décimale d'un entier positif a écrit en binaire. On sait que, par exemple

$$(1011)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 2 + 1 = 11$$

et on constate qu'on peut écrire cette suite de produits et d'additions en n'utilisant que deux opérations très simples, la multiplication par 2 et l'addition de 1 ou 0, sous la forme suivante [1] :

$$(1011)_2 = 2(2(2((2 \times 1) + 1) + 1) + 0) + 1$$

On voit donc que si on symbolise une variable, nulle au début, par le symbole $\square$, le calcul ci-dessus se fait en considérant a comme chaîne de caractères et en itérant l'opération

$$\square = 2 * \square + b$$

pour la valeur b (égale à 1 ou 0) de chaque caractère car de la chaîne a (lue de gauche à droite)

Ecrire alors une suite d'instructions construisant une fonction d'argument a (considéré comme une chaîne de caractères) et renvoyant la valeur de a en base décimale, puis expérimenter cette fonction par divers essais.

Exercice 2 : conversion décimal vers binaire par les plus grandes puissances

Ecrire une suite d'instructions permettant de convertir un entier (positif) de la base décimale vers son écriture binaire, en utilisant l'algorithme des plus grandes puissances (de 2) soustraites successivement.

Exercice 3 : conversion décimal vers binaire par divisions successives

Ecrire une suite d'instructions permettant de convertir un entier (positif) de la base décimale vers son écriture binaire, en utilisant l'algorithme de divisions successives par 2 vu en cours.

Exercice 4 : conversion décimal vers hexadécimal

Ecrire une suite d'instructions permettant de convertir un entier (positif) de la base décimale vers son écriture hexadécimale, en utilisant l'algorithme de divisions successives par 16.

Exercice 5 : square and multiply

L'algorithme de l'exponentiation rapide (appelé « square and multiply » en anglais) consiste à calculer le nombre x^n en utilisant l'écriture de l'entier n en binaire, ce qui conduit à un nombre d'opérations beaucoup moins important que le calcul de base $x^n = \underbrace{x \times x \ldots \times x}_{n \text{ fois}}$.

Si $n = (d_p d_{p-1} \ldots d_2 d_1 d_0)_2$ (où les d_i sont donc des 1 ou des 0), cela signifie que

$$n = d_0 + 2d_1 + 2^2 d_2 + \ldots + 2^{p-1} d_{p-1} + 2^p d_p$$

et donc

$$x^n = x^{d_0 + 2d_1 + 2^2 d_2 + \ldots + 2^{p-1} d_{p-1} + 2^p d_p} = x^{d_0} \times (x^2)^{d_1} \times (x^{2^2})^{d_2} \times \ldots \times (x^{2^{p-1}})^{d_{p-1}} \times (x^{2^p})^{d_p}$$

---

1. appelée « écrtiure de Hörner ».

Cet algorithme s'appuie sur l'égalité suivante, qui n'est qu'une autre façon de présenter ce qui précède :

$$x^n = \begin{cases} (x^2)^{\frac{n}{2}} \text{ si } n \text{ est pair} \\ x(x^2)^{\frac{n-1}{2}} \text{ si } n \text{ est impair} \end{cases}$$

On voit alors que, si n est pair, $d_0 = 0$ donc $x^{d_0} = 1$; il suffit alors de remplace n par $\frac{n}{2}$ et x par x^2 pour passer à l'examen de d_1. Si n est impair, alors $d_0 = 1$ et donc $x^{d_0} = x$; il suffit alors de remplacer n par $\frac{n-1}{2}$ et x par x^2 pour passer à l'examen de d_1. Et ainsi de suite.

On peut donc procéder au moyen d'une boucle qui, à chaque étape, rempacera x par x^2, et n par $\frac{n}{2}$ ou $\frac{n-1}{2}$ selon sa parité, tant que $n \geq 1$. Pour que cet algorithme calcule x^n, on peut introduire une variable réelle **puissance** de valeur initiale 1, qu'on multipliera à chaque étape par x si n est impair et qui restera inchangée si n est pair.

Ecrire alors une suite d'instructions qui calcule x^n par cette méthode, les valeurs x et n étant fournies par l'utilisateur.

Exercice 6 : conversion en binaire signé

Dans cet exercice, on suppose qu'on travaille avec N bits (par exemple, $N = 8$ pour simplifier).

Ecrire un algorithme de conversion d'un entier (de signe quelcconque) écrit en base décimale vers son écriture en binaire signé (c'est-à-dire avec complément à 2). Cet algorithme devra rejeter un nombre dont la conversion en binaire signé n'est pas réalisable sur N bits.

Exercice 7 : produit en binaire

Ecrire une suite d'instructions qui calcule le produit de deux nombres écrits en binaire fournis par l'utilisateur.