Aix*Marseille université

Portail et PES 1 René Descartes

Introduction à l'informatique : Partiel
Code UE : S14IN1I1

Année 2020-21

Documents et calculatrice interdits

Tous les exercices sont indépendants et peuvent donc être traités dans n'importe quel ordre. Le sujet est long (et le barème de correction en tiendra compte) : vous pouvez donc passer une question vous semblant trop difficile, quitte à y revenir ensuite s'il vous reste du temps. Par ailleurs, toute réponse non correctement rédigée (avec explication précise et concise) ou non justifiée sera considérée comme fausse.

Exercice 1 (codage des nombres)

1. Donnez le codage en binaire de l'entier naturel $(338)_{10}$, en précisant brièvement (5 lignes maximum) votre méthode.
2. Donnez l'entier naturel en base 10 dont le code binaire est $(11010111)_2$, en précisant brièvement (5 lignes maximum) votre méthode.
3. Donnez en base 10 le nombre réel en virgule flottante codé en binaire par les 32 bits suivants : 01011001 11101100 00000000 00000000.

Exercice 2 (nombres parfaits)

Soit n un nombre entier naturel. Le nombre $d \in \mathbb{N}$ est un diviseur de n si $n \bmod d = 0$, autrement dit si le reste de la division de n par d est nul. Un nombre entier naturel est *parfait* s'il est égal à la somme de ses diviseurs stricts (différents de lui-même). L'algorithme suivant permet de déterminer si un nombre entier naturel est parfait :

```
Fonction nb_parfait(n: entier): booléen
Début
   Si (n = 0) alors
      retour faux;
   Sinon
      s := 1;
      d := 2;
      Tant que d ≤ ⌊n/2⌋ faire      # ⌊n/2⌋: quotient de la division euclidienne de n par 2
         Si (n mod d = 0) alors
            s := s + d;
         Fin si
         d := d + 1;
      Fin faire
      Si (n ≠ s) alors
          retour faux;
      Sinon
          retour vrai;
      Fin si
   Fin si
Fin
```

1. Exécutez l'algorithme `nb_parfait` sur l'entier $n = 8$, en montrant toutes les valeurs des variables pendant l'exécution, le résultat et en comptant le nombre de tours de boucle (ou étapes d'itération) effectués. L'entier $n = 8$ est-il parfait ?
2. Même question pour l'entier $n = 28$.
3. Expliquez pourquoi l'algorithme `nb_parfait` termine pour toute entrée $n \geq 0$.
4. Calculez, en justifiant, le nombre d'opérations élémentaires effectuées par l'algorithme `nb_parfait` en fonction de la valeur de n, prise quelconque. Vous pourrez simplifier le résultat en utilisant la notation « grand $\mathcal{O}$ ».

Exercice 3 (palindromes)

Un palindrome est un mot dont la première lettre est identique à la dernière, la deuxième à l'avant-dernière, etc. À titre d'exemple, le mot « non », tout comme « esse » ou encore « ressasser » sont des palindromes de la langue française. Au delà de la langue française, le concept de palindrome s'étend à toutes les langues, mais aussi à toute suite de symboles d'un alphabet donné. Ainsi, le nombre entier 153 en base

10 devient un palindrome en base 2 : 10011001. La question « Was it a car or a cat I saw ? », est également un palindrome si l'on ne considère ni les espaces ni le point d'interrogation. Dans cet exercice, nous voulons écrire un algorithme qui décide si une chaîne de caractères représentée par un tableau de caractères est un palindrome, c'est-à-dire un algorithme qui répond vrai ou faux pour un mot donné en paramètre.

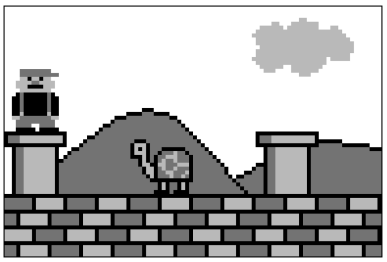
1. Écrivez en pseudo-code cet algorithme, dont le prototype est le suivant :

   `Fonction palindrome(C[`$0..n-1$`]: tab_caractères): booléen`,

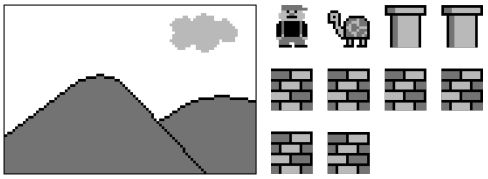
   où C$[0..n-1]$ représente un tableau de caractères de taille n. Prenez soin d'expliquer le principe de votre algorithme, en argumentant sa pertinence dans un texte court (une dizaine de lignes) le précédant.
2. Exécutez l'algorithme `palindrome` sur le mot « ressasser » en montrant toutes les comparaisons effectuées.
3. Évaluez la complexité en temps de votre algorithme. Vous calculerez pour ce faire précisément le nombre d'opérations élémentaires que vous simplifierez en utilisant la notation « grand $\mathcal{O}$ ».

Exercice 4 (codage d'informations)

Voici un écran du jeu vidéo *Super Plombiers Italiens* :



Cet écran consiste en un fond de 96 pixels de largeur et 64 pixels de hauteur, sur lequel sont superposés des *sprites*, c'est-à-dire des images carrées de 16×16 pixels qui représentent les objets du jeu, dans ce cas le plombier Marco, son ennemi la tortue, les deux tuyaux et 6 groupes de briques :



Cette version du jeu n'utilise que quatre couleurs (noir, gris foncé, gris clair et une couleur transparente, qui permet notamment de voir le fond derrière les personnages) pour les graphismes.

1. Combien de bits sont-ils nécessaires pour représenter le fond d'écran ? Combien d'octets ?
2. Combien de bits sont-ils nécessaires pour représenter chaque sprite ? Combien d'octets ?

Pour chaque écran du jeu, l'ordinateur a besoin de stocker les images du fond et de chaque sprite, mais ce n'est pas nécessaire de maintenir plusieurs copie de l'image de chaque sprite dans la mémoire s'il apparaît plusieurs fois : on peut tout simplement garder trace des coordonnés sur l'écran de chaque occurrence d'un sprite (plus spécifiquement, des cordonnées du point en bas à gauche de chaque sprite). Dans l'écran d'exemple, des briques identiques se trouvent aux coordonnés $(0, 0)$, $(16, 0)$, $(32, 0)$, $(48, 0)$, $(64, 0)$, et $(80, 0)$; deux tuyaux identiques se trouvent aux coordonnés $(0, 16)$ et $(64, 16)$. Marco se trouve aux coordonnées $(0, 32)$ et la tortue aux coordonnées $(32, 16)$.

3. Combien de bits sont-ils nécessaires pour représenter les coordonnés d'un sprite ? Combien d'octets ?

Donc, au delà du fond et d'un exemplaire de chaque image de sprite, on n'a besoin de stocker qu'une séquence de données de la forme (*numéro de sprite*, *coordonnés du sprite*) pour pouvoir reconstruire l'écran. Supposons que le jeu *Super Plombiers Italiens* comprenne un total de 256 sprites différents, qui représentent les protagonistes, les ennemis, les objets bonus, etc.

4. Combien d'octets sont-ils nécessaires pour représenter l'écran d'exemple (y compris le fond et les sprites) ?
5. De combien de mémoire (en kilo-octets) l'ordinateur doit-il disposer au minimum si chaque écran de jeu peut contenir un maximum de 16 sprites ?