



ICC

Examen Semestre I

Instructions :

- Vous disposez de 3 heures pour faire cet examen (8h00 - 11h).
- Nombre maximum de points : 90
- Attention : il y a aussi des énoncés sur le verso.
- Vous devez **écrire à l'encre noire ou bleu foncée**, pas de crayon ni d'autre couleur. N'utilisez **pas non plus de stylo effaçable** (perte de l'information à la chaleur).
- Toute documentation papier est autorisée ; En revanche, vous ne pouvez pas utiliser d'ordinateur personnel, ni de téléphone portable, ni aucun autre matériel électronique.
- Répondez sur les feuilles qui vous sont distribuées et **aux endroits prévus**. **Ne répondez pas sur l'énoncé.**
- Ne joignez aucune feuilles supplémentaires ; **seules les feuilles de réponses distribuées seront corrigées.**
- Vous pouvez répondre aux questions en français ou en anglais.
- L'examen compte 6 exercices indépendants (3 pour la théorie et 3 pour la programmation).

Vous pouvez commencer par celui que vous souhaitez

Exercice	1	2	3	4	5	6
Points	17	8	15	6.5	30 (7 + 12 + 6 + 5)	13.5

Exercice 1 : QCM Partie théorie [17 points]

Pour chaque question à choix multiple vous pouvez obtenir p points. Pour des réponses partiellement correctes, vous recevrez une fraction des points selon la formule « $p \cdot \max(0, x_C/C - x_I/I)$ », où C est le nombre de d'options correctes et I le nombre d'options incorrectes et x_C et x_I sont le nombre d'options correctes ou incorrectes que vous avez choisies respectivement. **Vous ne recevrez jamais de point négatifs pour une question.**

QCM 1.1 : Circuits [2 points]

Pour rappel, nous utilisons les symboles illustrés dans la figure 1 pour représenter les portes logiques ET (AND), OU (OR) et NON (NOT), respectivement.

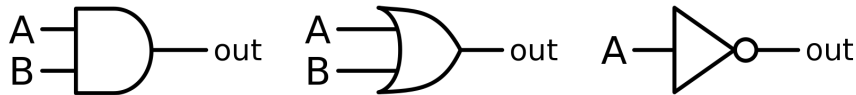


FIGURE 1 – Symboles d'une porte ET (AND, à gauche), d'une porte OU (OR, au milieu) et d'une porte NON (NOT, à droite).

Considérez le circuit de la figure 2 et sélectionnez **toutes les affirmations correctes**.

- A) Lorsque $D = 1$, la sortie X du circuit est toujours égale à 1.
- B) Lorsque $A = 0$, la sortie X du circuit est toujours égale à 0.
- C) Lorsque $C = 1$ et $D = 0$, la sortie X du circuit est égale à A .
- D) Lorsque $B = 1$ et $D = 0$, la sortie X du circuit est égale à C .

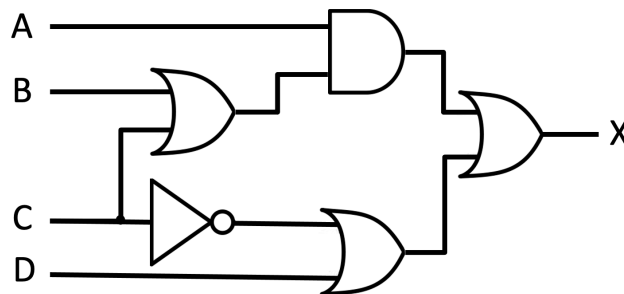


FIGURE 2 – Circuit

QCM 1.2 : Langage assembleur [3 points]

Supposons que l'on ait un CPU avec dix registres (r0 à r9) et les instructions de bases données dans la Table 1.

TABLE 1 – Instructions de base

Instruction	Signification
copy r0, c	$r0 \leftarrow c$: copie une constante c dans le registre r0
copy r0, r1	$r0 \leftarrow r1$: copie la valeur du registre r1 dans le registre r0
add r0, r1, c	$r0 \leftarrow r1 + c$: additionne une constante c à la valeur du registre r1 et enregistre le résultat dans le registre r0
add r0, r1, r2	$r0 \leftarrow r1 + r2$: additionne les valeurs des registres r1 et r2 et enregistre le résultat dans r0
multiply r0, r1, r2	$r0 \leftarrow r1 \cdot r2$: multiplie les valeurs des registres r1 et r2 et enregistre le résultat dans r0
divide r0, r1, r2	$r0 \leftarrow r1/r2$: division entière du registre r1 par r2 ; le résultat est enregistré dans r0
jump n	va à la ligne n du programme
jump_gt r0, r1, n	va à la ligne n si la valeur dans r0 est supérieure à la valeur dans r1 ($r0 > r1$)
stop	stoppe le programme

Considérez le programme en assembleur ci-contre à droite. Quelle est la valeur de r6 à la fin de l'exécution du programme, si r0 est initialisé à 3, r1 est initialisé à 2, et r2 est initialisé à 10 ?

- A) 18
- B) 19
- C) 20
- D) 21

```

1: copy r6, 1
2: copy r3, 1
3: jump_gt r3, r0, 12
4: copy r4, 1
5: jump_gt r4, r1, 9
6: copy r5, r4
7: add r4, r4, 1
8: jump 5
9: multiply r6, r6, r5
10: add r3, r3, 1
11: jump 3
12: add r6, r6, r2

```

QCM 1.3 : Échantillonnage et reconstruction [2 points]

Supposons que l'on veuille échantillonner le signal $X(t) = 5 \cdot \sin(100\pi t + \pi) + 10 \cdot \sin(12\pi t - \pi/2) + 3 \cdot \sin(50\pi t)$. Laquelle parmi les fréquences f_e suivantes est la plus petite fréquence qui permet une reconstruction correcte du signal $X(t)$ en utilisant la formule d'interpolation $X_I(t) = \sum_n X(nT_e) \text{sinc}(\frac{t-nT_e}{T_e})$ avec $T_e = 1/f_e$.

- A) $f_e = 50$ Hz
- B) $f_e = 100$ Hz
- C) $f_e = 200$ Hz
- D) $f_e = 300$ Hz

QCM 1.4 : Filtres [2 points]

Supposons que nous utilisons le signal $X(t) = 2 \cdot \sin(8\pi t) + 4 \cdot \sin(12\pi t)$ comme entrée d'un filtre passe-bas idéal avec une fréquence de coupure de f_{c1} et nous utilisons $\hat{X}_1(t)$ pour désigner le signal de sortie de ce filtre passe-bas idéal. Supposons en outre que $X(t)$ soit également utilisé comme entrée d'un filtre à moyenne mobile avec une fréquence de coupure de f_{c2} . Le signal de sortie de ce filtre à moyenne mobile est désigné par $\hat{X}_2(t)$. **Sélectionnez toutes les propositions correctes.**

- A) Si $f_{c1} > 6$, alors $X(t) = \hat{X}_1(t)$.
- B) Si $f_{c1} = f_{c2}$, alors $\hat{X}_1(t) = \hat{X}_2(t)$.
- C) Si $f_{c1} > f_{c2}$, alors la bande passante (bandwidth) de $\hat{X}_1(t)$ est toujours supérieure ou égale à $\hat{X}_2(t)$.
- D) Si $f_{c1} < f_{c2}$, alors la bande passante (bandwidth) de $\hat{X}_1(t)$ est toujours inférieure ou égale à $\hat{X}_2(t)$.

QCM 1.5 : Effet stroboscopique [3 points]

Supposons que l'on échantillonne le signal $X(t) = \sin(2\pi t) + \sin(4\pi t) + \sin(8\pi t)$ avec la fréquence d'échantillonnage $f_e = 6$ Hz et que nous reconstruisons le signal $X(t)$ en utilisant la formule d'interpolation $X_I(t) = \sum_n X(nT_e) \text{sinc}(\frac{t-nT_e}{T_e})$ avec $T_e = 1/f_e$. Sélectionnez les fréquences qui apparaissent dans le signal reconstruit $X_I(t)$. (Sélectionnez l'une des options ci-dessous.)

- A) 1 Hz et 2 Hz
- B) 1 Hz, 2 Hz, et 4 Hz
- C) 2 Hz, 4 Hz, et 6 Hz
- D) 2 Hz, 4 Hz, et 8 Hz

QCM 1.6 : Entropie [3 points]

Considérez les quatre mots suivants et ordonnez-les en fonction de leur entropie en commençant par le mot ayant la plus petite entropie jusqu'au mot ayant la plus grande entropie.

$w_1 = \text{abbreviation}$
 $w_2 = \text{facelessness}$
 $w_3 = \text{obfuscations}$
 $w_4 = \text{bacchanalian}$

Quel est l'ordre correct ?

- A) $H(w_2) < H(w_4) < H(w_1) < H(w_3)$
- B) $H(w_4) < H(w_2) < H(w_1) < H(w_3)$
- C) $H(w_2) < H(w_4) < H(w_3) < H(w_1)$
- D) $H(w_4) < H(w_2) < H(w_3) < H(w_1)$

QCM 1.7 : Spectre [2 points]

La figure 3 représente le spectre de fréquence (les lignes rouges) de l'un des signaux suivants. Sélectionnez le signal $X(t)$ à partir duquel ce spectre est dérivé.

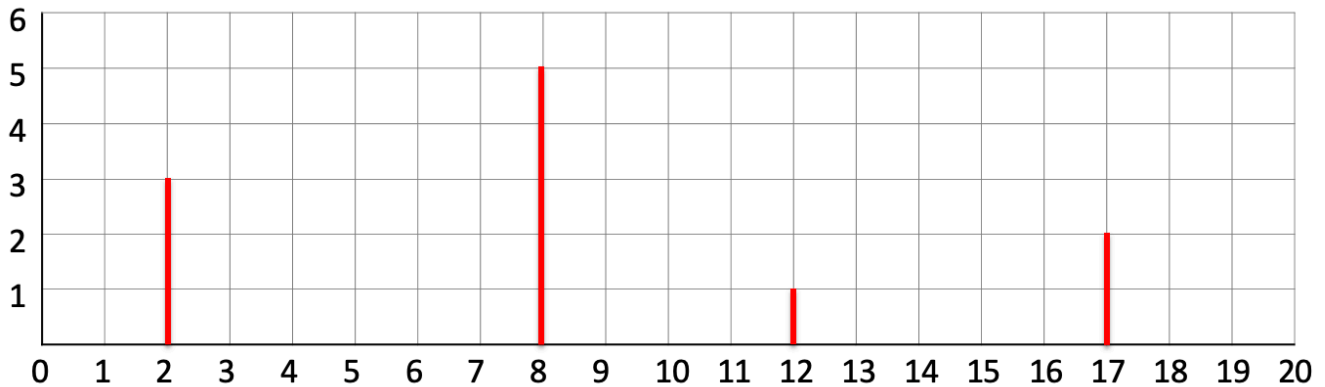


FIGURE 3 – Spectrum

- A) $X(t) = 3 \cdot \sin(2\pi t) + 5 \cdot \sin(8\pi t) + \sin(12\pi t) + 2 \cdot \sin(17\pi t)$
- B) $X(t) = 6 \cdot \sin(2\pi t) + 10 \cdot \sin(8\pi t) + 2 \cdot \sin(12\pi t) + 4 \cdot \sin(17\pi t)$
- C) $X(t) = 3 \cdot \sin(4\pi t) + 5 \cdot \sin(16\pi t) + \sin(24\pi t) + 2 \cdot \sin(34\pi t)$
- D) $X(t) = 6 \cdot \sin(4\pi t) + 10 \cdot \sin(16\pi t) + 2 \cdot \sin(24\pi t) + 4 \cdot \sin(34\pi t)$

Exercice 2 : Filtre à moyenne mobile [8 points]

Dessinez (sur la feuille réponse) le signal de sortie $\hat{X}(t)$ d'un filtre à moyenne mobile de période de coupure $T = 1$ s qui prend comme signal d'entrée le signal périodique $X(t)$ donné à la figure 4. La période du signal d'entrée $X(t)$ est de 4s.

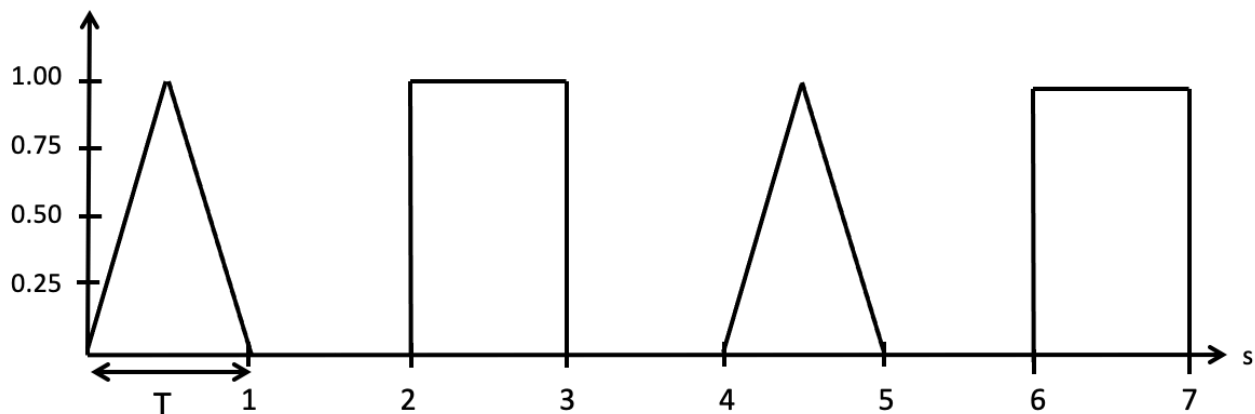


FIGURE 4 – Input signal $X(t)$ of a moving average filter with cutoff period T .

Exercice 3 : Codage de Huffman's [15 points]

Supposez que vous vouliez envoyer la phrase suivante (sans les espaces) à votre ami en utilisant le nombre minimal de bits.

ENGAGE LE JEU QUE JE LE GAGNE

Pour créer un encodage optimal avec l'algorithme de Huffman et trouver le nombre minimal de bits, procédez de la manière suivante :

- 3.1 Créez un tableau avec le nombre d'occurrences pour chaque lettre (voir le tableau sur la feuille réponse.)
- 3.2 Dessinez un arbre de Huffman pour ce tableau.
- 3.3 Créez un tableau avec un encodage pour chaque lettre (voir le tableau sur la feuille réponse.)
- 3.4 Calculez le nombre de bits nécessaires pour envoyer ce message avec votre encodage.
- 3.5 Combien de bits est-ce que l'on économise avec cet encodage par rapport à un encodage ASCII qui utilise 8 bits par lettre ?

Exercice 4 : Syntaxe C++ [6.5 points]

Pour chacune des instructions suivantes, indiquez si elle correspond à une instruction correcte ou incorrecte en C++. Dans le cas où c'est correct, expliquez brièvement à quoi l'instruction peut correspondre (initialisation, déclaration etc.) :

1. `char c("n");`
2. `vector <vector <int>> v(0,"init");`
3. `f(2) == 3;`
4. `3 = f(2);`
5. `void f = 3;`
6. `f((f(2) == 3));`
7. `constexpr int v(2);`
8. `f(double 3);`
9. `double f(3);`
10. `double f(double);`

Exercice 5 : Conception de programmes et programmation [30 points]

Vos services sont sollicités pour gérer une base de données d'empreintes digitales.



Une *empreinte digitale* est caractérisée par le nom de la personne à qui elle appartient (chaîne de caractères) ainsi que par une image en noir et blanc de *pixels*. Un pixel peut être représenté par une valeur booléenne (par exemple `true` pour noir et `false` pour blanc). Les images d'empreintes digitales sont des tableaux de taille fixe de 100x100 pixels. On suppose qu'il est possible de calculer la similarité de deux empreintes selon un algorithme connu mais dont les détails ne nous intéressent pas. Cette mesure est retournée sous la forme d'un entier positif, plus il est grand et plus les empreintes sont similaires. Deux empreintes sont similaires selon un seuil donné, `threshold`, si la mesure de leur similarité est supérieure ou égale à `threshold`.

Une *base de données d'empreintes digitales* est caractérisée par l'année où elle a été constituée, un identifiant (chaîne de caractères) et un ensemble d'empreintes digitales.

Question 1 : Structures de données/types [7 points]

Donner un code C++ possible pour les types/structures de données permettant de modéliser : un pixel, une image d'empreinte digitale, une empreinte digitale et une base de données d'empreintes digitales.

Les définitions des types/structures de données seront données dans l'ordre de précedence exigé par C++. Vous pourrez bien sûr définir d'autres types si nécessaire et ferez un usage judicieux du `typedef`.

Question 2 : Fonctionnalités [12 points]

Les fonctionnalités suivantes sont nécessaires à la gestion de nos bases de données d'empreintes digitales. Donnez les prototypes de ces fonctions. On ne vous demande pas d'écrire un programme complet ou le corps des fonctions mais uniquement les prototypes. Vous pouvez ajouter d'autres fonctions si vous estimez que c'est pertinent pour une bonne modularisation. Il n'est pas demandé non plus d'écrire des directives d'inclusion (`#include<...>`).

1. afficher une empreinte (son nom et l'image associée) ;
2. afficher une base de données d'empreinte digitale (son identifiant, année et toutes ses empreintes) ;
3. calculer la mesure de similarité de deux empreintes digitales données ;
4. vérifier si deux empreintes digitales données sont similaires selon un seuil de comparaison donné ;
5. créer une base de donnée vide (sans empreintes) en prenant en paramètre son identifiant et son année de création ;
6. ajouter une empreinte, lue depuis un fichier de nom donné, dans une base de données donnée (on supposera que le fichier contient le nom de la personne à qui appartient l'empreinte ainsi que les valeurs booléennes constituant l'image) ;
7. trouver (sans l'afficher), dans une base de données donnée, l'empreinte `e1` ayant la plus grande similarité avec une empreinte donnée `e2` et retourner le degré de similarité entre `e1` et `e2`.

Question 3 : Structures de données et fonctions (suite) [6 points]

Supposons maintenant que l'on s'intéresse à coder les deux fonctionnalités suivantes (on suppose que dans les recherches évoquées on ne compare pas une empreinte avec elle-même) :

8. Trouver sans les afficher toutes les paires d'empreintes digitales similaires selon un seuil donné, dans une base de données donnée ;
9. Étant donnée une liste d'empreintes digitales, trouver sans l'afficher, pour chacune d'elles, quelle autre empreinte de la liste est similaire selon un seuil donné. **Il se peut que pour un empreinte donnée, il n'existe pas d'empreinte suffisamment similaire.** Par exemple, étant donnée la liste { e1, e2, e3, e4, e5}, on cherche à obtenir une liste telle que {{e1,e4},{e2, e5}, {e3, x}, {e4, e1}, {e5,e2}}, signifiant que e1 est similaire à e4, e2 à e5 etc. Notez que x doit être remplacé par une information indiquant qu'aucune empreinte suffisamment similaire n'est trouvée.

Quel type additionnel (un seul) proposez-vous pour représenter la notion de *paire* d'empreintes digitales et qui satisfait aux besoins des deux fonctionnalités ci-dessus.

Proposez alors les prototypes des deux fonctionnalités additionnelles exprimées au moyen de ce nouveau type.

Question 4 : Programmation [5 points]

Donnez le code mettant en œuvre la fonctionnalité 8.

Souvenez vous que l'on ne compare pas une empreinte avec elle-même. Vous pourrez considérer que le nom de la personne à qui appartient l'empreinte identifie l'empreinte de façon unique. Vous utiliserez les fonctionnalités prototypées sans avoir besoin d'en donner les corps. Veillez à respecter une bonne indentation.

Exercice 6 : Déroulement de programme [13.5 points]

Le programme suivant compile et s'exécute sans erreurs (C++11).

```

1  #include <vector>
2  #include <iostream>
3  using namespace std;
4
5  typedef vector <int>  L;
6  typedef vector <L>  M;
7  constexpr int MIN_VALUE(-1000);
8
9  void d(ostream& out, const L& lst) {
10     for (const auto& v : lst){
11         out << v << " " << flush;
12     }
13     out << endl;
14 }
15
16 void d(ostream& out, const M& m){
17     for (const auto& ls : m){
18         d(out,ls);
19     }
20 }
21
22 double m(const L& lst){
23     double v(MIN_VALUE);
24     for (auto i : lst){ if (i > v) v=i; }
25     return v;
26 }
27
28 void m(const L& lst, int& idx){
29     idx=-1;
30     for (size_t i(0); i < lst.size(); ++i) {
31         if (lst[i] == m(lst)) { idx = i; return; }
32     }
33 }
34
35 int r(L& lst) {
36     int i;
37     m(lst, i);
38     if (i>=0){
39         lst[i] = lst[lst.size()-1];
40         lst.pop_back();
41     }
42     return i;
43 }
44 //SUITE DU CODE SUR LA PAGE SUIVANTE
45
46
47

```

```

48
49 L r(M& m){
50     L res;
51     for (auto& l : m){ res.push_back(r(l)); }
52     return res;
53 }
54
55 M ra(M& m){
56     M res;
57     while (m.size() != m[0].size()){ L sub(r(m)); res.push_back(sub); }
58     return res;
59 }
60
61 int main(){
62     M m3({{ 1, 1, 5, 1, 2, 1},
63           { 2, 1, -4, 1, 1, 1},
64           { 1, 8, 2, 1, 1, 1},
65           { 2, 1, 19, 1, 1, 1}});
66
67     M m4(m3); int k(0);
68
69     cout << ++k << "-----" << endl;
70     d(cout, m3);
71     cout << ++k << "-----" << endl;
72     r(m3);
73     d(cout, m3);
74     cout << ++k << "-----" << endl;
75     d(cout, m4);
76     cout << ++k << "-----" << endl;
77     d(cout, ra(m4));
78     cout << ++k << "-----" << endl;
79     d(cout, m4);
80     return 0;
81 }

```

1. Que font conceptuellement les fonctions `d` ?
2. Que font conceptuellement les fonctions `m` ?
3. (bonus) Quelle critique pouvez vous émettre sur l'efficacité du code mis en oeuvre pour coder la fonction de la ligne 28 ? décrivez brièvement en français comment il est possible d'améliorer cela ?
4. Que font conceptuellement les fonctions `r` ?
5. À la ligne 51 peut-on supprimer le symbole `&` en préservant la fonctionnalité initialement codée ? justifiez brièvement.
6. Que fait conceptuellement la fonction `ra` ?
7. Qu'affiche le programme ? Expliquez succinctement son déroulement. Il ne s'agit pas ici de paraphraser le code, mais bien d'*expliquer* les étapes et le déroulement du programme.