

CORRIGÉ ÉPREUVE REGROUPEE

Discipline: INFORMATIQUE

Date: 19/12/2023

1. Dans l'Histoire plus ou moins ancienne, citer un ancêtre des ordinateurs actuels.

(3 pts)

En donner une brève description et décrire en quoi il a joué un rôle important.

Ex: Le récipient avec des cailloux qu'utilisaient des bergers préhistoriques pour comptabiliser les moutons sortis de l'enclos. Il permettait de stocker cette information hors du cerveau humain et ce stockage pouvait perdurer très longtemps.

Autres exemples d'ancêtres: pascaline / machine à tisser de Jacquard / Machine analytique de Babbage / Eniac / etc.

2. Expliquer l'utilité de trois types de mémoire

(3 pts)

Choisir trois types de mémoire présents dans la pyramide (aide-mémoire) et donner un exemple d'utilisation montrant l'utilité de ce type de mémoire.

- type 1: Registres

C'est le stockage le plus rapide, le seul qui permet au processeur de lire une instruction, des données et d'écrire le résultat d'un calcul en un seul tic d'horloge.

- type 2: mémoire cache

Mémoire plus rapide que la RAM dans laquelle on peut copier une portion du programme en train de tourner sur le processeur. Comme ça, on a pas mal de chances de pouvoir charger les instructions suivantes suffisamment rapidement pour ne pas faire patienter le processeur. Si une instruction n'est pas dans la mémoire cache, il faut la chercher sur la RAM et là, le temps est nettement plus long.

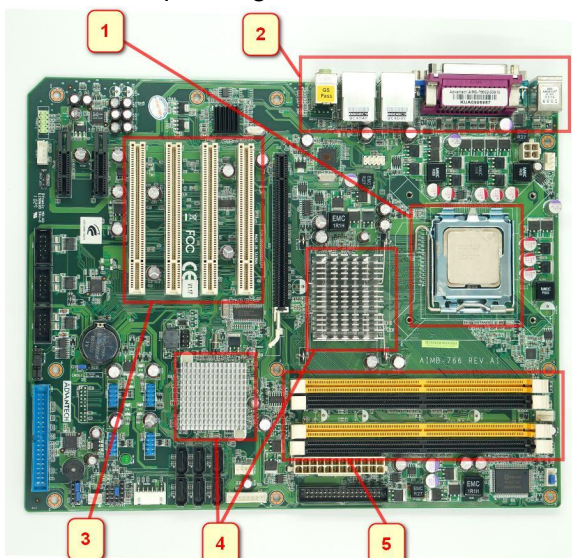
- type 3: RAM

Mémoire centrale où se trouvent tous les programmes qui sont "en train" de tourner sur le processeur. En réalité, seul l'un d'entre eux (ou plutôt le nombre correspondant au nombre de coeurs du processeur) tourne réellement sur le processeur (ou sur l'un de ses noyaux) mais les autres sont en attente et passent régulièrement sur le processeur. Quand ils ont besoin d'accéder à une donnée qui n'est pas présente sur le cache (RAM, voire stockage ou internet), l'ordonnanceur de l'OS fait passer un autre processus.

- etc.

3. Parmi les éléments suivants, **en choisir 2** qui sont encadrés sur l'image ci-dessous et indiquer, à gauche de leur nom, le nombre relié à leur cadre:

(2 pts)



- **5** RAM (mémoire vive)
- **1** CPU (processeur)
- **4** Chipset (North, Southbridge)
- **3** PCI (slots d'extension)
- **2** Entrées / Sorties(périphériques)
- ~~boîtier (carcasse de l'ordinateur)~~

4. Parmi les deux processeurs de smartphones ci-dessous, indiquer celui (3 pts)

qui est capable de faire le plus de calculs par seconde d'après les indications disponibles. Indication: "Cores" se traduit en français par "Cœurs".

A		Qualcomm Snapdragon 800 2,27 GHz
	Model	Qualcomm MSM 8974
	Cores	4
B		Qualcomm Snapdragon 665 2.02 GHz
	Model	Qualcomm Technologies, Inc SM6125
	Cores	8

Réponse (à justifier):

A: peut faire théoriquement $4 \times 2,27 = 9,08$ milliards de calculs / s (GHz)

B: peut faire théoriquement $8 \times 2,02 = 16,16$ milliards de calculs / seconde

Donc, **B** est plus puissant car sa fréquence d'horloge est plus basse mais il a deux fois plus de cœurs.

5. Chaque capsule a besoin de 300 Mo (en moyenne) pour être stockée. (3 pts)

Sachant que le stockage total occupe 12,3 Go:

a. Combien de groupes de 3^{ème} année y a-t-il? (2 pts)

12,3Go = 12 300Mo. Or, $12\,300\text{Mo} / 300\text{Mo} = 41$. Il y a donc **41 capsules**.

b. Combien de bits contient une capsule en moyenne? (1 pt)

1 octet = 8 bits, donc $300\text{Mo} = 2400\text{Mb}$ ou **2,4Gb, donc 2,4 milliards de bits**

6. Binaire... (5 pts)

a. Qu'est-ce qu'un bit? (1 pt)

réponse: c'est un **chiffre binaire**, autrement dit un chiffre qui peut valoir "0" ou "1".

C'est aussi la plus petite unité de stockage.

b. Écrire le nombre binaire **0100 1011**₍₂₎ en système décimal (base 10). (1 pt)

réponse: $0 \times 128 + 1 \times 64 + 0 \times 32 + 0 \times 16 + 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1 = 75$ ₍₁₀₎

c. Écrivez le nombre décimal **113**₍₁₀₎ en système binaire (base 2). (1 pt)

réponse: 113 ₍₁₀₎ = $0 \times 128 + 1 \times 64 + 1 \times 32 + 1 \times 16 + 0 \times 8 + 0 \times 4 + 0 \times 2 + 1 \times 1 = 111\,0001$ ₍₂₎

d. Combien de bits faut-il pour coder les 4 directions (Nord, Sud, Est, Ouest)? (1 pt)

réponse: **2** car on peut écrire 4 nombres différents (00/01/10/11), suffisants donc.

e. Proposer un code binaire pour chaque point cardinal de la question précédente. (1 pt)

réponse: arbitraire: **00 = Nord / 01 = Sud / 10 = Est / 11 = Ouest**

7. Représentation des caractères (4 pts)

a. Comment fait-on pour stocker dans un ordinateur des caractères alphanumériques (1 pt)

comme ceux présents dans cette épreuve alors que l'ordinateur ne comprend que 1/0?

réponse: On attribue à chaque caractère alphanumérique un code binaire, par ex. avec le **tableau ASCII**.

b. En utilisant le codage correspondant à la question a, numériser le texte suivant: (3 pts)

Mdr ! réponse: **0100 1101 / 0110 0100 / 0111 0010 / 0010 0000 / 0010 0001**

8. Décoder le texte présent sur la carte perforée ci-dessous... (3 pts)



Texte décodé:

colonne 1: 0101 0100 → T

colonne 2: 0111 0101 → u

colonne 3: 0111 0010 → r

colonne 4: 0110 1001 → i

colonne 5: 0110 1110 → n

colonne 6: 0110 0111 → g

Donc, le texte est : **Turing**

9. Algorithmes efficaces ou pas...

(3 pts)

- a. Comment s'appelle la stratégie algorithmique consistant à calculer toutes les possibilités d'un problème avant d'y donner une solution ?

(1 pt)

Force brute

- b. Expliquer pourquoi un programme correct obéissant à la stratégie mentionnée ci-dessus peut ne pas donner de solution quand on l'exécute sur un ordinateur avec des données d'entrée de taille trop grande.

(2 pts)

Car le nombre de calculs est si immense que le temps nécessaire à exécuter l'algorithme est irréalisable. Par exemple, si le temps est plus grand que l'âge de l'Univers, ça n'est d'aucune utilité!

10. Voici un algorithme en pseudo-code.

(8 pts)

```
DEBUT
VARIABLE compteur ← 1
DEMANDER nombre
TANTQUE compteur < nombre
{
  AVANCER_DE compteur
  TOURNER_DROITE_DE 90°
  AUGMENTER_DE_1 compteur
}
FIN
```

Dessin ci-dessous:



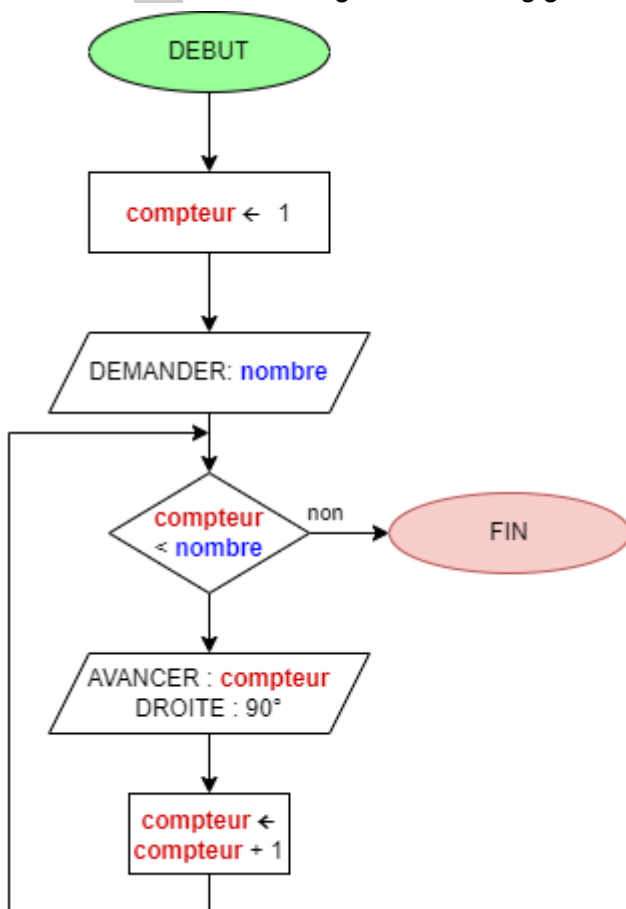
- a. Dessiner à droite de l'algorithme ce qu'il produit sachant que l'utilisateur a choisi la valeur 5 pour **nombre**:

(~3 pts)

L'échelle du dessin est libre et on considère que l'orientation au départ pointe vers la droite.

- b. Traduire l'algorithme en logigramme:

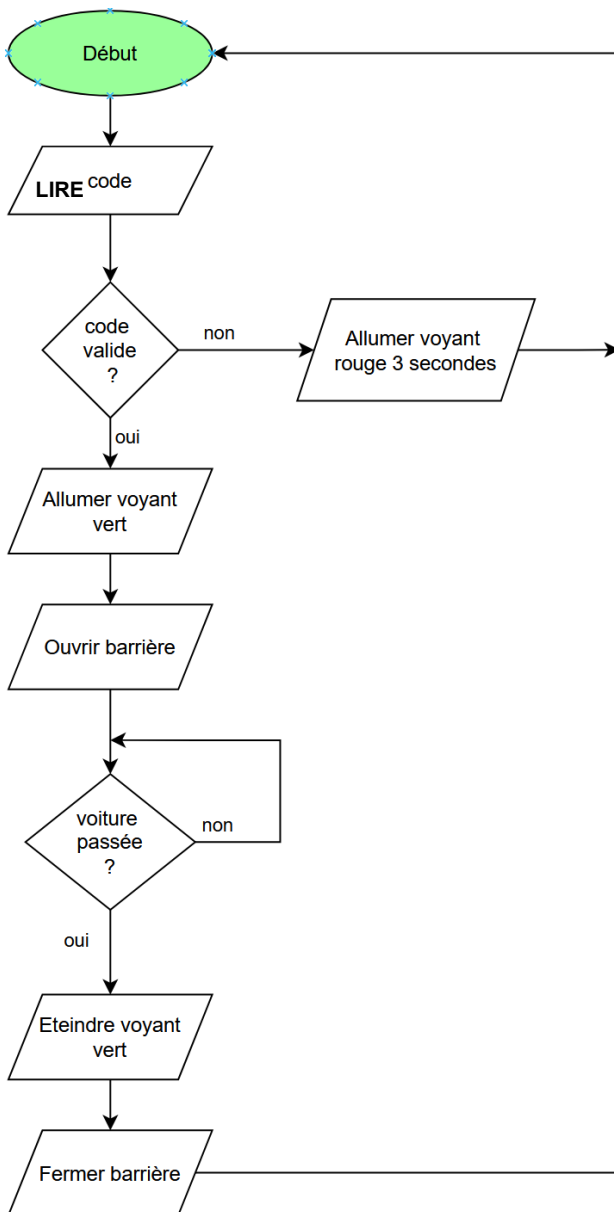
(~5 pts)



11. Voici un algorithme en logigramme.

(8 pts)

Logigramme:



a. Décrire ce que fait cet algorithme. (2 pts)

Il gère l'ouverture d'une barrière à l'entrée d'un parking où les abonnés ont des codes pour entrer.

b. Traduire en pseudo-code. (4 pts)

DEBUT:

LIRE code

SI code valide

{

ALLUMER voyant vert

OUVRIR barrière

TANTQUE voiture pas passée

{RIEN}

ETEINDRE voyant vert

FERMER barrière

SAUTER_VERS **DEBUT**

}

SINON

{

ALLUMER voyant rouge pendant 3s

SAUTER_VERS **DEBUT**

}

c. Que se passe-t-il si un conducteur saisit un code valide mais fait demi-tour et s'en va ? (1 pt)

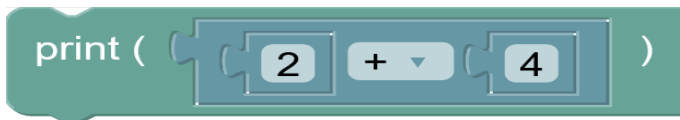
La **barrière reste ouverte** et le voyant vert allumé.

d. Comment améliorer cela ? (1 pt)

On peut modifier la condition du **TANTQUE** en ajoutant une clause sur le temps écoulé comme: **TANTQUE** (voiture pas passée) **ET** (temps<30s) ainsi, dès que le temps dépasse 30s, la barrière se ferme et le voyant s'éteint, comme si une voiture était passée.

12. Voici un programme en Blockly / Python:

(1 pt)



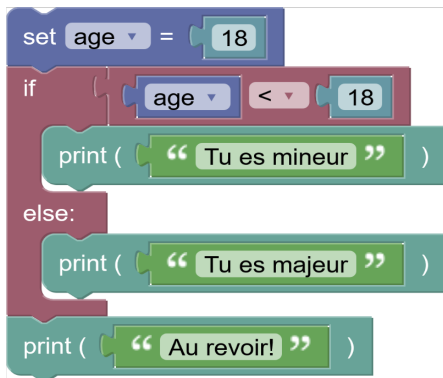
```
1 print(2+4)
2
```

Qu'est-ce qu'affiche ce programme ?

réponse: 6

13. Voici un programme en Blockly / Python:

(3 pts)



```

1 age = 18
2 if age < 18:
3     print("Tu es mineur")
4 else:
5     print("Tu es majeur")
6 print("Au revoir!")
  
```

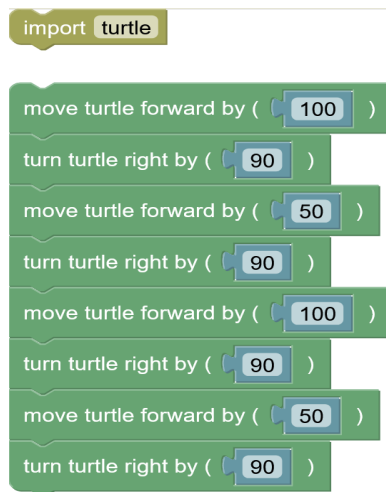
Qu'est-ce qu'affiche exactement ce programme ?

Tu es majeur

Au revoir!

14. Voici un programme en Blockly / Python qui fait un dessin.

(6 pts)



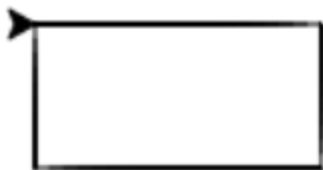
```

1 import turtle
2
3 turtle.forward(100)
4 turtle.right(90)
5 turtle.forward(50)
6 turtle.right(90)
7 turtle.forward(100)
8 turtle.right(90)
9 turtle.forward(50)
10 turtle.right(90)
  
```

a.

Dessiner la figure produite par ce programme.

(2 pts)



b.

Réécrire le programme (en bloc ou Python) pour éviter les répétitions de code..

(4 pts)

Programme modifié ci-dessous:

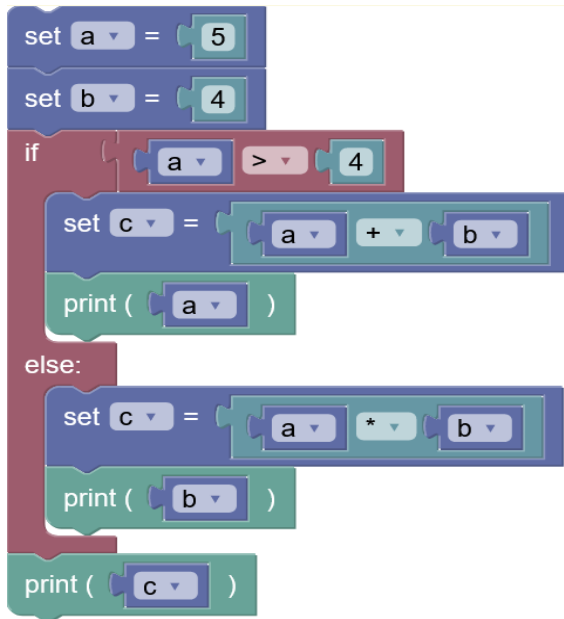


```

1 import turtle
2 for i in range(0, 2):
3     turtle.forward(100)
4     turtle.right(90)
5     turtle.forward(50)
6     turtle.right(90)
7
  
```

15. Voici un programme en Bloc / Python :

(3 pts)



```

1 a = 5
2 b = 4
3 if a > 4:
4     c = a + b
5     print(a)
6 else:
7     c = a * b
8     print(b)
9 print(c)

```

a.

Quelles sont les valeurs des variables ci-dessous à la fin de l'exécution du programme?

(1 pts)

a: 5 b: 4

b.

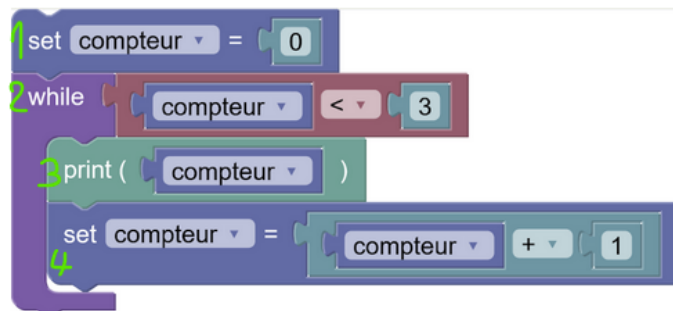
Qu'est-ce qu'affiche ce programme à l'écran exactement ?

(2 pts)

5
9

16. Voici un autre programme en Bloc / Python :

(4 pts)



```

1 compteur = 0
2 while compteur < 3:
3     print(compteur)
4     compteur = compteur + 1

```

Il affiche ceci à l'écran: 0 / 1 / 2 (chaque nombre sur une ligne nouvelle)

a.

(2 pts)

Modifier le moins possible le programme (indiquer comment ci-dessous) pour qu'il affiche ceci:

1 Modification: ligne1: compteur = 1
2

b.

(2 pts)

Modifier le moins possible le programme (indiquer comment ci-dessous) pour qu'il affiche ceci:

0 Modification: ligne3: print(compteur * 3)
3
6

FIN DE L'ÉPREUVE