

ÉPREUVE REGROUPEE

Discipline: INFORMATIQUE

Date: 19/12/2023

Cours: 1 IN DP 06 / 07 / 08 / 09 / 10

Durée: 105 minutes

Maître: JC Barros

Matériel autorisé:

- **Aucun (aide-mémoire en fin d'épreuve)**

Consignes de travail:

- **Ecriture lisible et propre demandée**
- **Mettre en évidence les résultats**
- **8 pages + aide-mémoire à la fin**

Nom et prénom:

Total: 62 points

-
- 1.** Dans l'Histoire plus ou moins ancienne, citer un ancêtre des ordinateurs actuels.
En donner une brève description et décrire en quoi il a joué un rôle important.

(3 pts)

- 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:

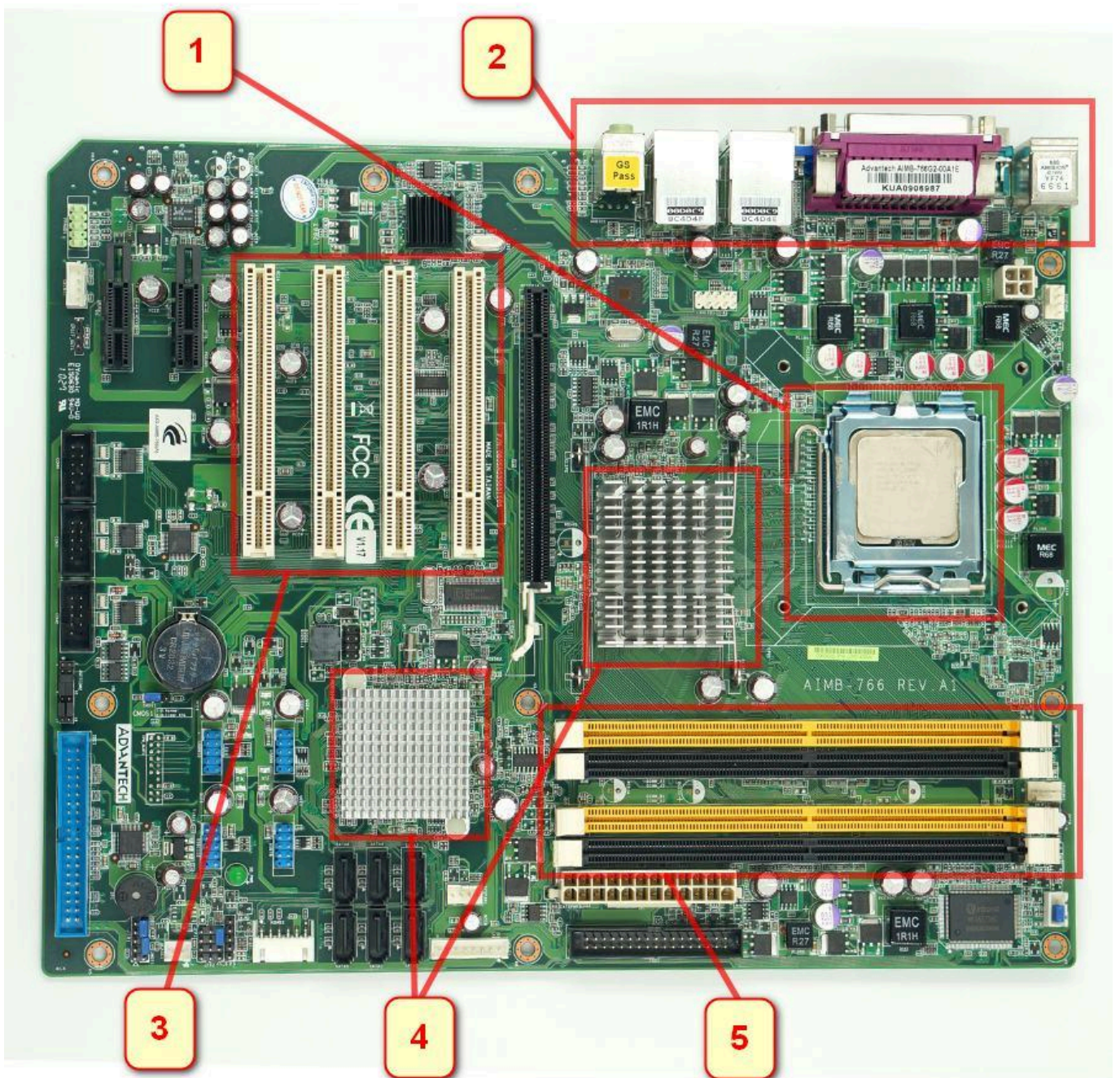
- type 2:

- type 3:

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)

- RAM (mémoire vive)
- CPU (processeur)
- Chipset (Northbridge, Southbridge)
- PCI (slots d'extension)
- Entrées / Sorties (connecteurs pour périphériques)
- boîtier (carcasse de l'ordinateur)



4. Parmi les deux processeurs de smartphones ci-dessous, indiquer celui qui est capable de faire le plus de calculs par seconde d'après les indications disponibles. (3 pts)
- Indication: "Cores" se traduit en français par "Coeurs".

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):

5. On veut stocker les capsules vidéo produites par des groupes de 3^{ème} année. (3 pts)
- Chaque capsule a besoin de 300 Mo (en moyenne) pour être stockée.
- Sachant que le stockage total occupe 12,3 Go:
- a. Combien de groupes de 3^{ème} année y a-t-il? (2 pts)
- Montrer le détail des calculs et donner une réponse avec une phrase complète.

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

6. Binaire... (5 pts)

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

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

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

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

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

7. Représentation des caractères

(4 pts)

- a. Comment fait-on pour stocker dans un ordinateur des caractères alphanumériques (lettres, chiffres, signes de ponctuation) comme ceux présents dans cette épreuve alors que l'ordinateur ne sait manipuler que des "1" et des "0" ?

(1 pt)

réponse:

- b. En utilisant le codage correspondant à la question a, numériser (c'est-à-dire traduire en nombres binaires) le texte suivant:

(3 pts)

Mdr ! (ne pas oublier l'espace avant le !)

réponse:

8. Décoder le texte présent sur la carte perforée ci-dessous...

(3 pts)

←

Carte perforée de 80 octets

	0	1	2	3	4	5	6	7	8	9	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40	42	44	46	48	50	52	54	56	58	60	62	64	66	68	70	72	74	76	78	80					
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8

←

Chaque colonne représente un octet. Chaque ligne x correspond à sa valeur 2^x.
Une position libre vaut 0. Une position perforée vaut 1.
La colonne 0 sert d'exemple. Elle indique le nombre 42.

Texte décodé:

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)

- 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.
Indication: Penser au problème du voyageur de commerce, par exemple...

(2 pts)

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.

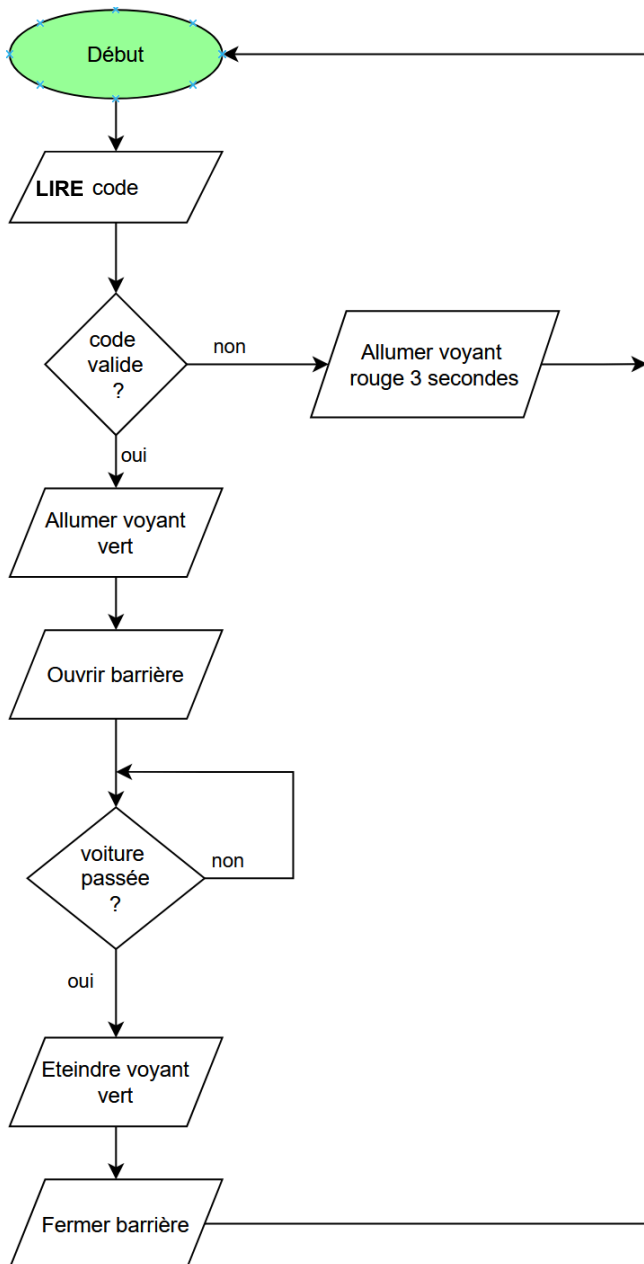
- b. Traduire l'algorithme en logigramme:

(~5 pts)

11. Voici un algorithme en logigramme.

(8 pts)

Logigramme:



a. Décrire ce que fait cet algorithme.

(2 pts)

b. Traduire en pseudo-code.

(4 pts)

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

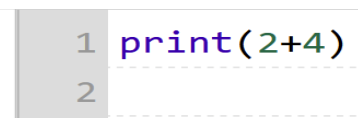
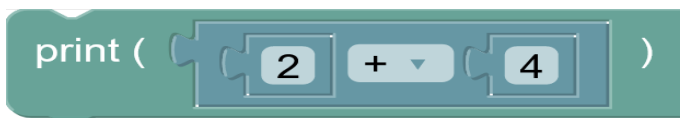
(1 pt)

d. Que faut-il modifier pour qu'il traite correctement ce cas ?

(1 pt)

12. Voici un programme en Blockly / Python:

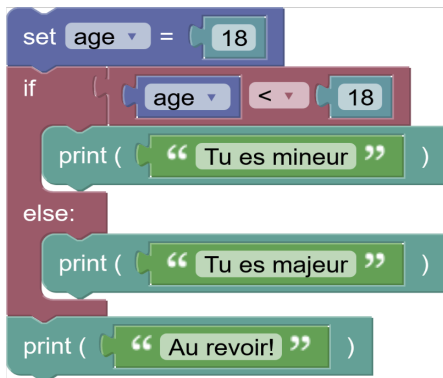
(1 pt)



Qu'est-ce qu'affiche ce programme ?

13. Voici un programme en Blockly / Python:

(3 pt)



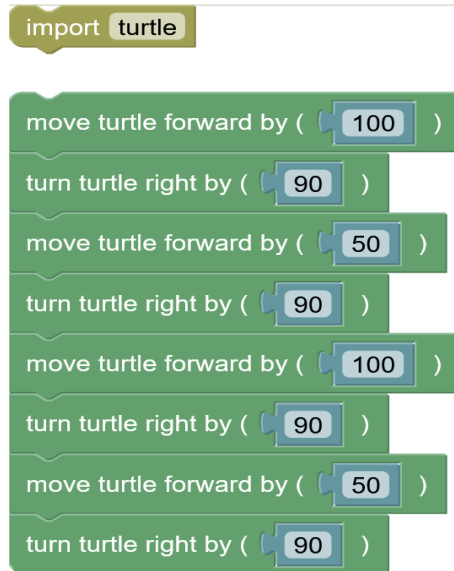
```

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 ?

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)

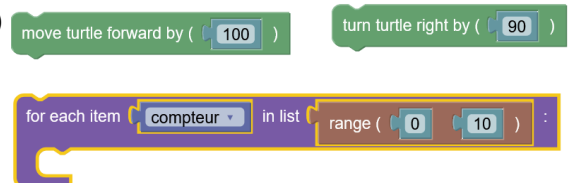
b.

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

(4 pts)

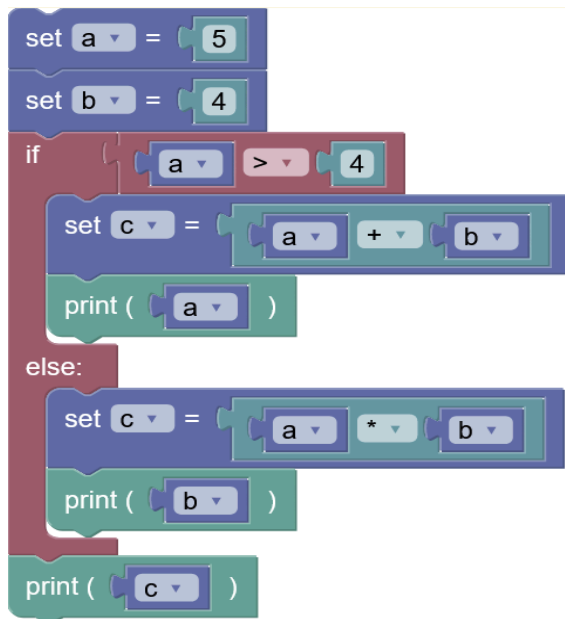
Il faut donc introduire une boucle... (exemples de blocs à ici:)

Programme modifié ci-dessous:



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:

b:

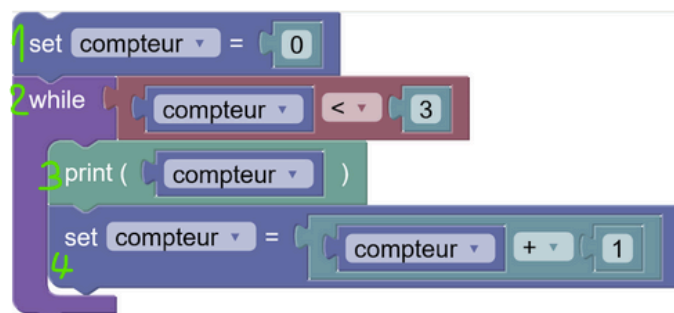
b.

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

(2 pts)

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

2

b.

(2 pts)

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

0

3

6

FIN DE L'ÉPREUVE

Aide-mémoire informatique

1^{ère} année - 1^{er} semestre

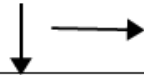
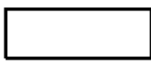
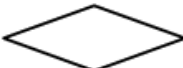
Préfixes:

Liste des préfixes très utilisés en informatique (où “o” signifie “octet”).:

To Go Mo ko o

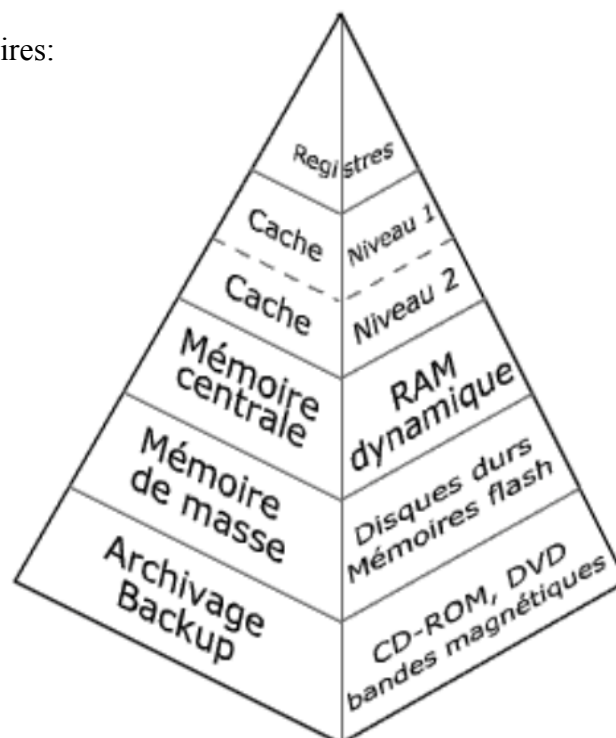
Algorithmes:

Voici les formes utilisables dans un logigramme:

Symbole	Nom	Rôle
	Flèche	Indique la séquence du programme
	Terminaison	Début ou termine un programme
	Processus	Exécute une tâche
	Décision	Génère une décision (oui, non, etc.)
	Entrée/Sortie	Accepte une entrée ou produit une sortie

Mémoires:

Voici la pyramide des mémoires:



ASCII TABLE

Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]	48	30	110000	60	0	96	60	1100000	140	`
1	1	1	1	[START OF HEADING]	49	31	110001	61	1	97	61	1100001	141	a
2	2	10	2	[START OF TEXT]	50	32	110010	62	2	98	62	1100010	142	b
3	3	11	3	[END OF TEXT]	51	33	110011	63	3	99	63	1100011	143	c
4	4	100	4	[END OF TRANSMISSION]	52	34	110100	64	4	100	64	1100100	144	d
5	5	101	5	[ENQUIRY]	53	35	110101	65	5	101	65	1100101	145	e
6	6	110	6	[ACKNOWLEDGE]	54	36	110110	66	6	102	66	1100110	146	f
7	7	111	7	[BELL]	55	37	110111	67	7	103	67	1100111	147	g
8	8	1000	10	[BACKSPACE]	56	38	111000	70	8	104	68	1101000	150	h
9	9	1001	11	[HORIZONTAL TAB]	57	39	111001	71	9	105	69	1101001	151	i
10	A	1010	12	[LINE FEED]	58	3A	111010	72	:	106	6A	1101010	152	j
11	B	1011	13	[VERTICAL TAB]	59	3B	111011	73	;	107	6B	1101011	153	k
12	C	1100	14	[FORM FEED]	60	3C	111100	74	<	108	6C	1101100	154	l
13	D	1101	15	[CARRIAGE RETURN]	61	3D	111101	75	=	109	6D	1101101	155	m
14	E	1110	16	[SHIFT OUT]	62	3E	111110	76	>	110	6E	1101110	156	n
15	F	1111	17	[SHIFT IN]	63	3F	111111	77	?	111	6F	1101111	157	o
16	10	10000	20	[DATA LINK ESCAPE]	64	40	1000000	100	@	112	70	1110000	160	p
17	11	10001	21	[DEVICE CONTROL 1]	65	41	1000001	101	A	113	71	1110001	161	q
18	12	10010	22	[DEVICE CONTROL 2]	66	42	1000010	102	B	114	72	1110010	162	r
19	13	10011	23	[DEVICE CONTROL 3]	67	43	1000011	103	C	115	73	1110011	163	s
20	14	10100	24	[DEVICE CONTROL 4]	68	44	1000100	104	D	116	74	1110100	164	t
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]	69	45	1000101	105	E	117	75	1110101	165	u
22	16	10110	26	[SYNCHRONOUS IDLE]	70	46	1000110	106	F	118	76	1110110	166	v
23	17	10111	27	[END OF TRANS. BLOCK]	71	47	1000111	107	G	119	77	1110111	167	w
24	18	11000	30	[CANCEL]	72	48	1001000	110	H	120	78	1111000	170	x
25	19	11001	31	[END OF MEDIUM]	73	49	1001001	111	I	121	79	1111001	171	y
26	1A	11010	32	[SUBSTITUTE]	74	4A	1001010	112	J	122	7A	1111010	172	z
27	1B	11011	33	[ESCAPE]	75	4B	1001011	113	K	123	7B	1111011	173	{
28	1C	11100	34	[FILE SEPARATOR]	76	4C	1001100	114	L	124	7C	1111100	174	
29	1D	11101	35	[GROUP SEPARATOR]	77	4D	1001101	115	M	125	7D	1111101	175	}
30	1E	11110	36	[RECORD SEPARATOR]	78	4E	1001110	116	N	126	7E	1111110	176	~
31	1F	11111	37	[UNIT SEPARATOR]	79	4F	1001111	117	O	127	7F	1111111	177	[DEL]
32	20	100000	40	[SPACE]	80	50	1010000	120	P					
33	21	100001	41	!	81	51	1010001	121	Q					
34	22	100010	42	"	82	52	1010010	122	R					
35	23	100011	43	#	83	53	1010011	123	S					
36	24	100100	44	\$	84	54	1010100	124	T					
37	25	100101	45	%	85	55	1010101	125	U					
38	26	100110	46	&	86	56	1010110	126	V					
39	27	100111	47	'	87	57	1010111	127	W					
40	28	101000	50	(	88	58	1011000	130	X					
41	29	101001	51	)	89	59	1011001	131	Y					
42	2A	101010	52	*	90	5A	1011010	132	Z					
43	2B	101011	53	+	91	5B	1011011	133	[					
44	2C	101100	54	,	92	5C	1011100	134	\					
45	2D	101101	55	-	93	5D	1011101	135	]					
46	2E	101110	56	.	94	5E	1011110	136	^					
47	2F	101111	57	/	95	5F	1011111	137	_					

Programmation (Python)

Remarque: Certaines instructions n'ont peut-être pas encore été vues en classe

Calculs :

opérations de base: + - * / **

reste et quotient de division euclidienne: % //

var % div vaut 0 si var est divisible par div

ex: 5 // 2 = 2 5 % 2 = 1

Variables et affectations :

types: int, float, str, bool, list ...

int(), float() pour **convertir** en int ou en float

affectation simple : ma_variable = -5

permutation de variables: a, b = b, a

affectations combinées: +=, -=, *=, etc.

ex: x += 1 est comme x = x + 1

Listes :

liste: L = [-1, 3.2, "abc"]

liste vide: []

multiplication: [0]*3 -> [0, 0, 0]

élément d'une liste: L[1] = 5

doit déjà exister, l'indice commence à 0.

ajouter un élément à une liste :

L.append(-2.5)

supprimer élément à la position i :

del L[i] ou L.pop(i)

supprimer élément de valeur v dans L :

L.remove(v)

longueur d'une liste L :

len(L)

somme des valeurs d'une liste:

sum(liste)

slice: L[i:j] (i inclus ! j exclu)

Chaînes de caractères :

chaîne: txt="bonjour" ou txt='bonjour'

concaténation: txt + " est un texte"

élément, slice d'une chaîne: texte[i]

comme listes mais non modifiable

Entrées/sorties :

sortie: print(valeur,... , end="\n")

valeur de type quelconque, end optionnel

entrée: texte = input("message")

Attention: retourne toujours un str

Commentaires :

sur une ligne : # texte

plusieurs lignes : '''texte''' ou """texte"""

Structures de contrôle :

1^{ère} ligne finit par ':' puis indentation des lignes du bloc

tests simples: ==, !=, <, <=, >, >=, in

tests combinés: and, or, not, a < b < c

conditionnelle

if condition1:

instructions

elif condition2:

elif = contraction de else if

instructions

else:

instructions

boucle while (boucle non bornée) :

while condition:

instructions

boucle for (boucle bornée) :

for i in liste: # boucle parcourt

la liste

instructions

for i in range(5): # pour i de 0 à 4

instructions

range(i,j,k): de i (inclus) à j (exclu) par pas de k

sauter, quitter une boucle : continue, break

Fonctions:

définition de la fonction f à deux paramètres:

def f(x, y):

reponse = x**2 + y**2

return reponse

appel de la fonction: f()

variable = f(3, 5)

print(f(2, 6))

etc.