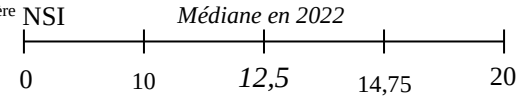
CORRIGE CONTROLE 1 (55') PROGRAMMATION DE BASE EN PYTHON

Compte rendu :

- ***Les exos sur France IOI et les quizz (note > 80 %) doivent être faits sinon à faire pendant l'évaluation !***
- ***Contrôle souvent mal préparé : Toutes les questions ont été faites soit dans les quizz soit dans les évaluations des années précédentes soit dans le livret !***
- *Programmation (exo 4) : Réussi très rarement (3/11 fois en 2022 ; 2/10 fois en 2021), souvent du n'importe quoi !*
  Les cas « particuliers » (branche a = 0) sont oubliés, mal traités ou carrément supprimés d'office. Voir correction.
  Un programme doit traiter tous les cas possibles d'entrée et pas seulement ceux qui sont faciles !

Plus généralement : Bien lire tous les détails de ce corrigé. Apprendre mieux le cours, refaire les quizz jusqu'à ce qu'ils soient réussis (> 80 %) et préparer des évaluations des années précédentes. Relisez-vous !

Médianes = 12,5 en 2022 ; 11,63 en 2021 ; 11 en 2020 ; 11,5 en 2019.



➢ Exercice n° 1 (……..…… / 6 points) : Syntaxe et opérations de base.

① Evaluer les expressions suivantes : (…… / 4 × 0,25 pts)

3 + 3 / 3 → *4.0* 13 % 3 → *1* '2' + '1' → *'21'* -3**2 → *-9*

Beaucoup d'erreurs dans les 2 derniers : 21 au lieu de '21' et erreur de signe au dernier.

Rappels : Opérateur « / » : quotient de la division classique (attention, renvoie forcément un float)

Opérateur « % » : reste dans la division pseudo-euclidienne (ou modulo), renvoie un int si les deux opérandes sont int, float sinon.

Opérateur « + » entre 2 str → concaténation (accolement) entre les deux str.

*Opérateur « ** » : puissance mathématique, renvoie un int si les deux opérandes sont int, float sinon.*

② Evaluer les expressions suivantes : (…… / 2 × 0,5 pts)

(-1)**2 == 1.0 → *True* 'A' > 'a' → *False*

Rappels : 1 == 1.0 est vrai *Dans la table Unicode, numéros des majuscules < numéros des minuscules.*

③ Le contraire de « (A et B) est vrai. » *équivaut à (A et B) est faux donc A faux ou B faux (1ère loi de Morgan) :*

L'un au moins de A et B est faux.	A et B sont tous les 2 faux.	L'un seulement de A et B est faux.	Aucun de A et B sont vrais.

④ Entourer l'expression qui renvoie True :

False or False and True False or True and False True or False and False True and False or False

Rappel : and > or. *Méthode : Mettre des parenthèses puis appliquer les résultats du and puis du or.*

⑤ Quel choix présente un accumulateur mais pas un compteur ou décompteur ?

Rappel : Un accumulateur est une variable incrémentée (auto-affection additive) dans une boucle. Un compteur ou décompteur est une variable incrémentée ou décrémentée de 1 dans une boucle.

compteur = 0 for k in range(5) : compteur = k + 1	compteur = 0 for k in range(5) : compteur=compteur+1	compteur = 0 for k in range(5) : compteur=compteur+k	compteur = 0 for k in range(5) : compteur=compteur – 1

⑥ La liste d'entiers de m inclus à n inclus équivaut à :

Rappel : range(début,fin) fabrique un itérable allant de début inclus à fin exclue.

range(m+1 , n+1)	range(m+1 , n)	range(m , n+1)	range(m – 1, n+1)

➢ Exercice n° 2 (……..…… / 4 points) : Structures de contrôle de base. (1 point par question)

① Parmi les scripts suivants, entourer **celui ou ceux** qui présentent une boucle **infinie** :

```
k = 0
while k**2 <= k :
    k = k + 1
```

Boucle finie à 2 tours car à partir de k = 2, k² > k. Donc la condition devient fausse.

```
k = 1 000 000
while k != 1 :
    k = k – 1
```

Boucle finie à 1 000 000 de tours.

```
for k in Séquence :
    Séquence.extend([0])
```

Voir cours Python2 p.5 E].

```
k = 0
while k < -1 or k > -2 :
    k = k – 1
```

Boucle infinie car l'une des 2 conditions dans le while toujours vraie.

② Soit N un entier ≥ 1

```
k = 0
while 2**k <= N :
    k = k + 1
```

Ecrire l'instruction à ajouter à la suite de ce script qui permettra d'afficher le plus grand exposant de la puissance de 2 contenue dans N :

Voir corrigé contrôle1 2021.

③

```
for k in range(2) :
    if k == 1 :
        break
    print('oui', end=' ')
else :
    print('non')
```

Peu réussi ! Beaucoup ne connaissent pas le comportement du else :
Else associé à une boucle For est une clause confirmative qui ne s'exécutera que si le break ne se déclenche pas.
Ici ce n'est pas le cas !

Le script à gauche affiche :

1) oui oui
2) oui oui non
3) oui
4) oui non

④ Soient les deux boucles imbriquées suivantes :

```
N = k = 0
while k <= 7 :
    while k < 5 :
        k = k + 1
        N = N + 1
    k = k + 1
```

Peu réussi. Faire un tableau d'états.

Compléter :

Nb de tours externes = *1 + 1 +1*

Nb de tours internes = *5 + 0 + 0*

valeur de N = *5*

valeur de k = *8*

➢ Exercice n° 3 (……..… / 4 points) : Structures de contrôle de base : Répéter … jusqu'à ce que.

La boucle « Répéter (faire) …. jusqu'à ce que …. » (repeat (do) … until…) existe dans certains langages (Pascal, LUA etc.) mais pas en Python.

Cette structure de contrôle se présente sous la forme ci-contre :

Répéter (faire) :

Bloc d'instructions

Jusqu'à ce que ⓒ

	Boucle Repeat Until	*Boucle While*
Condition	*Arrêt*	*Poursuite*
Ordre d'exécution	*Corps – Condition.*	*Condition – Corps.*
Nombre d'exécutions du corps de boucle	*Toujours au moins 1 fois.*	*Peut être 0 fois.*
	Autant de fois que la vérification de la condition.	*1 fois de moins que la vérification de la condition.*

1. D'après le schéma, ⓒ est : (entourer la bonne réponse) (……. / 1 pt) *S'aider du tableau précédent.*

une variable.	la négation de la condition d'arrêt de la boucle.	la condition d'arrêt de la boucle.	la condition de poursuite de la boucle.

2. Entourer la ou les affirmations vraies : (……. / 1 pt) *S'aider du tableau précédent.*

Le bloc est exécuté une fois de plus que la condition.	Le bloc est exécuté 1 fois au moins.	La condition est vérifiée 1 fois de plus que l'exécution du bloc.	La boucle Repeat Until est équivalente à la boucle While.

Il est évident à la vue des différences dans le tableau précédent qu'une boucle Repeat Until n'est pas équivalente à une boucle While, même si leurs comportements sont proches.

3. Soit le script (écrit en pseudo-Python) :

```
n = 0
repeat :
        n = n + 1
        if n % 2 == 1 :
            print( 2**n, end='' )
until n >= 4
```

a. Qu'affiche exactement ce script ? *2 8* (…… / 1 pt)

Le corps de boucle va s'exécuter de n=0 à n=3 soit 4 fois, et affichage en ligne des puissances de 2 seulement pour les n impairs. Très peu réussi !

b. Réécrire la ligne qu'il faut modifier pour afficher 4 16 (…….. / 1 pt)

4 = 2^2 et 16 = 2^4 donc il faut afficher les puissances paires de 2 et donc sélectionner les puissances paires dans la condition du test. Très peu réussi !

➢ Exercice n° 4 (……..… / 6 points) : Un grand classique (exo 1 p.12 livret de cours Python 2).

Soit l'équation $aX^2 + bX + c = 0$.

1. Ecrire un programme qui : (…… / 5 points)
   - en entrée reçoit les valeurs des 3 coefficients a, b et c.
   - en sortie renvoie la ou les solutions s'il y en a, ou la phrase 'Pas de solution !' sinon.

   Attention à bien traiter tous les cas possibles suivant le triplet (a , b , c).

Voir corrigé du contrôle 1 de 2020.

2. Pour s'assurer du bon fonctionnement du programme, il faut tester 4 cas bien choisis.

   Quels sont ces 4 cas ? Les entourer. (……... / 1 pt)

a = 0	b = 0	a ≠ 0	b ≠ 0	a = 0 et c = 0	a = 0 et b = 0 et c = 0	a = 0 et b = 0 et c ≠ 0	a = 0 et b ≠ 0