26. Un entier positif est *premier* s'il ne peut pas être divisé que par lui-même et par 1. Par exemple, 7 est un nombre premier parce qu'il est divisible seulment par 1 et 7. Est-ce que la fonction suivante est correcte pous tous les entiers $n \geq 2$?

```
def premier(n):
    '''(int)->bool
    Retourne True si n est premier, et False sinon.
    Precondition: n est un entier positif'''

    est_premier=True
    for i in range(2,n):
        if n%i == 0:
            est_premier=False
    return est_premier
```

(a) Non. Elle n'est pas correcte quand n est 2.
(b) Non. Pour la corriger, après la ligne `est_premier=False` ligne, on doit ajouter le fragment de code suivant et aligner le `else` avec le `if`.

```
    else:
        est_premier=True
```

(c) Non. Pour la corriger, la première ligne de la fonction doit être remplacée par `est_premier=False` et la quatrième ligne par `est_premier=True`
(d) Non. Pour la corriger, `range(2,n)` doit être remplacée par `range(2,n+1)`;
(e) Oui.

27. La fonction `produit_vectoriel` permet de calculer le produit factoriel de deux listes. Elle prend deux listes des entiers (de la même taille) et retourne leur produit vectoriel. Le produit vectoriel de deux listes $x = [x_1, x_2, ..., x_n]$ et $y = [y_1, y_2, ..., y_n]$ est la somme $x_1y_1 + x_2y_2 + ... + x_ny_n$. Laquelle des implémentations suivantes est la solution correcte?

I)
```
def produit_vectoriel(a, b):
    '''(list,list)->int'''
    pv=0
    for i in range(len(a)):
        for j in range(len(b)):
            pv=pv+a[i]*b[j]
    return pv
```

II)
```
def produit_vectoriel(a, b):
    '''(list,list)->int'''
    pv = 0
    for i in range(len(a)):
        pv = pv + a[i]*b[i]
    return pv
```

III)
```
def produit_vectoriel(a, b):
    '''(list,list)->list'''
    pv=[]
    for i in range(len(a)):
        pv.append((a[i]*b[i]))
    return pv
```

(a) I (b) II (c) III (d) Plusieurs des I, II, III (e) Aucune de I, II, III

22. Pour la fonction ci-dessous, lequel des appels suivants donne erreur?

```
def mostar(s):
    i = 0
    resultat = ""
    while not s[i].isdigit():
        resultat = resultat + s[i]
        i = i + 1
    return resultat
```

(a) `mostar('123')`
(b) `mostar('abc123')`
(c) `mostar('123abc')`
(d) `mostar('abc')`
(e) Plusieurs des appels ci-dessus.

23. Qu'est-ce que le programme Python suivant va afficher sur l'écran? ?

```
def format_nom(fn, ln, ap):
    print(ln+" "+fn+" "+ap)

format_nom('Theodor', 'Seuss', 'Dr.')
```

(a) `'Seuss Theodor Dr.'`
(b) `('Seuss Theodor Dr.')`
(c) `Seuss Theodor Dr.`
(d) `'Dr. Seuss Theodor'`
(e) `None`

24. Qu'est-ce que le programme Python suivant va afficher sur l'écran? ?

```
def format_nom(fn, ln, ap):
    print(ln+" "+fn+" "+ap)

r=format_nom('Theodor', 'Seuss', 'Dr.')  + 'est un auteur.'
```

(a) `'Seuss Theodor Dr. est un auteur.'`
(b) `('Seuss Theodor Dr. est un auteur.')`
(c) `Seuss Theodor Dr. est un auteur.`
(d) `'Dr. Seuss Theodor est un auteur.'`
(e) Donne erreur.

25. Que fait la fonction suivante?

```
def kiki(s):
    '''(str) -> bool
    Precondition: len(s) est un entier positif
    '''
    count = 0
    for i in range(len(s) // 2):
        if s[i] == s[len(s)//2 + i]:
            count = count + 1
    return count == (len(s) // 2)
```

(a) Retourne True si la première moitié de s est la même que la deuxième moitié renversée et False sinon.
(b) Retourne True si la première moitié de s est la même que la deuxième moitié et False sinon.
(c) Retourne True si la première moitié de s n'est pas la même que la deuxième moitié et False sinon.
(d) Retourne True si le premier élément est égal au dernier et False sinon.
(e) Aucune des déclarations ci-dessus.

20. La fonction encryp permet d'encrypter une chaine de caractères entrée en paramètre(la description détaillée est ci-dessous). Une ligne de code manque dans le corps de cette fonction (voir le commentaire #ligne manque). Quelle ligne doit-on y ajouter pour que la fonction soit correcte?

```
def encrypt(s):
    '''str ->str
    Retourne une nouvelle chaine de caracteres dont le premier et le deuxieme caractere de s devient le av
    le troisieme et le quatrieme caractere de s devient le quatrieme et le troisieme de la fin;
    et on continue s'il y a encore de caracteres.
    Outremont dit', la premiere paire de caracteres de s devienne la derniere paire;
    la deuxieme paire devienne la paire avant la derniere, etc.

    Precondition: len(s) >= 2, len(s) % 2 == 0

    >>> encrypt('ab')
    'ab'
    >>> encrypt('1234abcd')
    'cdab3412'
    '''
    sn=""
    for i in range(0,len(s),2):
        # ligne manque
    return sn
```

(a) sn=sn+s[i]+s[i+1]
(b) sn=sn+s[i+1]+s[i]
(c) sn=s[i]+s[i+1]+sn
(d) sn=sn[::-1] + s[i]+s[i+1]
(e) Plusieurs des solutions ci-dessus sont correctes.

21. Vous devez implémenter une fonction donne_cartes qui donne num cartes au joueur s'il y a num cartes dans le paquet; sinon, elle donne toutes les cartes qui restent dans le paquet (les cartes sont enlevées de la liste paquet et mis dans la liste joueur). Quelle ligne doit-on y ajouter au lieu de #ligne manque pour résoudre ce problème?

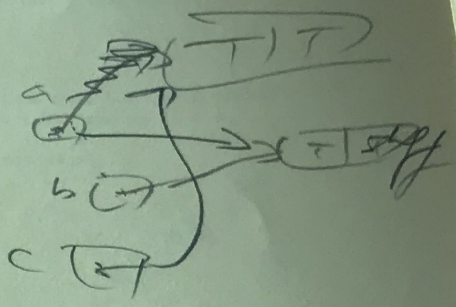
```
def donne_cartes(paquet, joueur, num):
    '''(list of int, list of int, int)-> None
    Donne num carte au joueur si len(paquet)>=num, sinon donne len(joueur) cartes.
    Precondition: num>=1, len(paquet)>=1
    >>> paquet=[2, 3, 2, 4, 2, 3]
    >>> joueur=[3, 3, 4]
    >>> donne_cartes(paquet, joueur, 4)
    >>> joueur
    [3, 3, 4, 3, 2, 4, 2]
    >>> paquet
    [201, 303]
    >>> paquet=[2, 3]
    >>> joueur=[3, 3, 4]
    >>> donne_cartes(paquet, joueur, 4)
    >>> joueur
    [3, 3, 4, 3, 2]
    >>> paquet
    []
    '''
    i=1
    #ligne manque
        joueur.append(paquet.pop())
        i=i+1
```

(a) while i<=num-1:
(b) while i<=num:
(c) while i<=num and len(paquet)!=0:
(d) while i<=num or len(paquet)<=0:
(e) for i in range(0,len(paquet),num):

17. Qu'est-ce que le programme Python suivant va afficher sur l'écran?

```
a=[True, True]
b=[True, True]
c=a
a=b
a[1]="stuff"
print(a,b,c)
```

(a) [True, 'stuff'] [True, 'stuff'] [True, True]
(b) [True, 'stuff'] [True, 'stuff'] [True,'stuff']
(c) [True, 'stuff'] [True, True] [True, True]
(d) [True, 'stuff'] [True, 'True'] [True,'stuff']
(e) Rien. Donne erreur.



18. Qu'est-ce que le programme Python suivant va afficher sur l'écran?

```
a=[True, True]
b=[True, True]
c=a
a=b
a=[True, 'stuff']
print(a,b,c)
```

(a) [True, 'stuff'] [True, 'stuff'] [True, True]
(b) [True, 'stuff'] [True, 'stuff'] [True,'stuff']
(c) [True, 'stuff'] [True, True] [True, True]
(d) [True, 'stuff'] [True, 'True'] [True,'stuff']
(e) Rien. Donne erreur.

19. La fonction ci-dessous permet de retourner le nombre maximal de monais de deux dollars et le nombre de monais de un dollar qui couvrent le montant donné comme paramètre (un entier). Quel code doit-on y ajouter pour que le corps de la fonction soit correcte?

```
def monais(montant):
    '''(int)->tuple of (int, int)
    Preconditions: amount>=0
    Retourne un tuple avec
    le nombre de monais des deux dollars
    et le nombre de monais de un dollar
    >>> monais(9)
    (4,1)
    >>> monais(12)
    (6, 0) '''
```

(a) `return (montant//2, montant%1)`
(b) `return (montant//2, montant%2)`
(c) `return (montant/2, montant%2)`
(d) `return (montant%2, montant//1)`
(e) `return (montant%2, montant//2)`

1 c
2 a
3 d
4 a
5 a
6 b
7 e
8 b
9 b
10 d
11 b
12 c
13 d
14 e
15 a
16 d
17 a
18 c
19 b
20 c
21 c
22 d
23 c
24 e
25 b
26 e
27 b