État d'Israël
Ministère de l'Éducation
Type d'examen : *Bagrout*
Session de l'examen : été 2019
Numéro de l'examen : 899381
Traduction française (4)

מדינת ישראל
משרד החינוך
סוג הבחינה: בגרות
מועד הבחינה: קיץ תשע"ט, 2019
מספר השאלון: 899381
תרגום לצרפתית (4)

Informatique

מדעי המחשב

Instructions au candidat

הוראות לנבחן

א. Durée de l'examen : trois heures.

א. משך הבחינה: שלוש שעות.

ב. Structure du questionnaire et barème de notation :
Ce questionnaire comporte trois sections.

Première section	— (1 × 10) + (1 × 15)	— 25 points
Deuxième section	— (2 × 25)	— 50 points
Troisième section	— (1 × 25)	— 25 points
	Total	— 100 points

ב. מבנה השאלון ומפתח ההערכה:
בשאלון זה שלושה פרקים.

פרק ראשון	– (1×10) + (1×15)	— 25 נקודות
פרק שני	– (2×25)	— 50 נקודות
פרק שלישי	– (1×25)	— 25 נקודות
	סה"כ	— 100 נקודות

ג. Matériel auxiliaire autorisé :
Tout matériel auxiliaire, à l'exception d'un ordinateur programmable.

ג. חומר עזר מותר בשימוש:
כל חומר עזר, חוץ ממחשב הניתן לתכנות.

ד. Instructions particulières :
(1) Tous les programmes qu'il vous est demandé d'écrire en langage informatique dans la première et deuxième section, doivent être écrits dans un seul langage – Java ou C#.
(2) **Inscrivez sur la couverture** de votre cahier d'examen dans quel **langage** vous écrivez – Java ou C#.
(3) **Inscrivez sur la couverture** de votre cahier d'examen le nom de la **filière dans laquelle vous avez étudié**. Cette filière est l'une des quatre suivantes : מודלים חישוביים, מבוא לחקר ביצועים, מערכות מחשב ואסמבלי, תכנות מונחה עצמים.
Remarque : dans les programmes que vous écrivez, vous ne perdrez pas de points si vous écrivez une majuscule au lieu d'une minuscule ou l'inverse.

ד. הוראות מיוחדות:
(1) את כל התכניות שאתה נדרש לכתוב בשפת מחשב בפרקים הראשון והשני כתוב בשפה אחת בלבד — Java או C#.
(2) **רשום על הכריכה החיצונית** של המחברת באיזו **שפה** אתה כותב — Java או C# .
(3) **רשום על הכריכה החיצונית** של המחברת את שם ה**מסלול שלמדת.** המסלול הוא אחד מארבעת המסלולים האלה: מערכות מחשב ואסמבלי, מבוא לחקר ביצועים, מודלים חישוביים, תכנות מונחה עצמים.
הערה: בתכניות שאתה כותב לא יורדו לך נקודות, אם תכתוב אות גדולה במקום אות קטנה או להפך.

Écrivez dans votre cahier d'examen uniquement. Indiquez "טיוטה" en haut de chaque page utilisée comme brouillon.
Écrire des notes de brouillon en dehors de celles du cahier d'examen, risque d'entraîner votre disqualification.

כתוב במחברת הבחינה בלבד. רשום "טיוטה" בראש כל עמוד המשמש טיוטה.
כתיבת טיוטה בדפים שאינם במחברת הבחינה עלולה לגרום לפסילת הבחינה.

Bonne chance!

בהצלחה!

Questions

Ce questionnaire comporte trois sections.
Répondez à des questions des trois sections,
en suivant les instructions énoncées dans chaque section.

Première section (25 points)

Remarque : pour chaque question où une saisie est requise, il n'est pas nécessaire de vérifier la validité de la saisie.

Pour ceux qui composent en Java : pour chaque question où une saisie est requise, supposez que le programme comporte l'instruction :

Scanner input = new Scanner (System.in);

Répondez à la question 1 – obligatoire (10 points).

1. Écrivez une méthode externe exact en Java ou Exact en C#, acceptant un tableau – arr de type chaîne et un nombre entier num . La méthode retournera le nombre de chaînes du tableau arr dont la longueur est égale à num .

Répondez à l'une des questions 2-3 (15 points).

2. Le système informatique d'un magasin de camping comporte une classe **Flashlight** comportant deux attributs :

model – nom d'un modèle de lampe-torche, de type chaîne.

price – prix de la lampe-torche de type réel.

א. Écrivez en Java ou en C# , la déclaration de la classe **Flashlight** ainsi que ses attributs, puis écrivez une méthode constructeur acceptant des valeurs pour le modèle de la lampe-torche ainsi que pour son prix.

ב. **(1)** En vue d'une randonnée, vous devez vous munir de trois lampes-torches différentes. Écrivez une méthode externe threeFlashlights en Java ou ThreeFlashlights en C# acceptant un tableau – arr d'objets différents de type **Flashlight** et un nombre – total de type réel.

La méthode devra trouver et afficher une seule fois, les modèles des trois lampes-torches dont la somme des prix est le nombre total .

Supposez que pour chaque attribut de la classe **Flashlight** , les méthodes get et set en Java et les méthodes Get et Set en C#, ont été définies.

Supposez qu'il existe une seule combinaison de trois lampes-torches dont la somme des prix est le nombre total .

(2) Quelle est la complexité d'exécution de la méthode que vous avez écrite au sous-paragraphe ב(1) ? Justifiez.

/suite en page 3/

3. Dans un système informatique d'une société de vente de voitures d'occasion, on a défini une classe **Car** comportant trois attributs :
licenseNum – numéro d'immatriculation de type chaîne.
hadAccident – variable de type booléenne acceptant true si la voiture a subi un accident, autrement – elle accepte false .
price – prix de la voiture de type entier.
Supposez qu'une méthode constructeur acceptant des valeurs pour chacun des attributs de la classe a été défini, et que pour chaque attribut, les méthodes get et set en Java et les méthodes Get et Set en C#, ont été définies.

א. Écrivez dans la classe **Car** , une méthode interne booléenne range en Java ou Range en C#, acceptant deux nombres de type entier min et max . La méthode retournera true si le prix de la voiture se situe entre min et max (inclus). Autrement, la méthode retournera false .
Supposez que max est supérieur à min .

ב. Soit la classe **AllCars** comportant deux attributs :
cars – un tableau unidimensionnel de type **Car** .
num – le nombre actuel de voitures qu'il y a dans la société, de type entier.
Voici la méthode constructeur de la classe.

```
public AllCars ( int max )
{
    this.cars = new Car [max];
    this.num = 0;
}
```

(1) Écrivez dans la classe **AllCars** , une méthode interne booléenne addCar en Java ou AddCar en C#, acceptant un élément de type **Car** . La méthode ajoute cet élément au tableau des voitures, au premier emplacement libre, puis retourne true .
Si le tableau est plein, la méthode retournera false .
Supposez qu'il n'y a pas d'emplacement libre du début du tableau jusqu'au nombre num (num non-inclus).

(2) Écrivez dans la classe **AllCars** , une méthode interne print en Java ou Print en C#, acceptant deux nombres min et max de type entier (max est supérieur à min).
La méthode affichera dans le tableau cars , le numéro d'immatriculation de toute voiture du tableau cars qui n'a pas subi d'accident et dont le prix se situe entre les deux nombres min et max (inclus).
Il est obligatoire d'utilisez la méthode définie au paragraphe א.

/suite en page 4/

Deuxième section (50 points)

Attention : Dans chaque question où une implémentation est requise, vous pouvez utiliser les méthodes des classes File, Pile, Arbre binaire et Chaîne sans les implémenter. Si vous utilisez des méthodes supplémentaires, implémentez-les.

Répondez à deux des questions 4-6 (25 points par question).

4. **Attention** : Cette question comporte deux versions : en Java (pages 4 et 5) et en C# (pages 6 et 7). Répondez dans le langage que vous avez étudié.

Pour ceux qui résolvent en Java

Soit la méthode secret1 :

```
public static boolean secret1 (int num, int digit )
    {
        if (num < 10)
            return ( num % 2 == digit % 2 );
        if ( num % 2 != digit % 2 )
            return false;
        return secret1 ( num /  10, digit) ;
    }
```

א. **(1)** Écrivez ce que retournera l'appel de la méthode secret1 (937, 5) . Montrez la trace.

(2) Donnez un exemple d'un nombre num à 3 chiffres, pour lequel l'exécution de la méthode secret1(num,5) retournera une valeur différente de celle obtenue au sous-paragraphe (1). Montrez la trace.

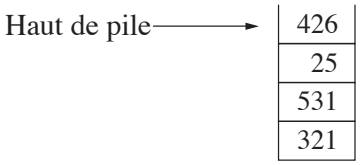
(3) Écrivez en une phrase, ce qu'exécutera la méthode booléenne secret1 , c'est à dire à quelle question la méthode retourne true ou false .

/suite en page 5/

Soit la méthode secret2 :

```
public static boolean secret2 ( Stack <Integer> s )
    {
        boolean ok;
        int x;
        if ( s.isEmpty () )
            ok = true ;
        else
        {
            x = s.pop () ;
            if ( ! (secret1 (x, x % 10)) )
                ok = false;
            else
                ok = secret2 (s);
        }
        return ok ;
    }
```

ב. **(1)** Pour la pile s ci-dessous :



Écrivez ce que retournera la méthode secret2 . Montrez la trace (il n'est pas nécessaire de montrez la trace de la méthode secret1).

(2) Écrivez en une phrase, ce qu'exécutera la méthode booléenne secret2 , c'est à dire à quelle question la méthode retourne true ou false .

/suite en page 6/

Pour ceux qui résolvent C# :

Soit la méthode Secret1 :

```
public static bool Secret1 (int num, int digit )
    {
        if ( num < 10 )
            return ( num % 2 == digit % 2 );
        if ( num % 2 != digit % 2 )
            return false;
        return Secret1 ( num / 10, digit) ;
    }
```

א. **(1)** Écrivez ce que retournera l'appel de la méthode Secret1 (937, 5) . Montrez la trace.

(2) Donnez un exemple d'un nombre num à 3 chiffres, pour lequel l'exécution de la méthode Secret1(num,5) retournera une valeur différente de celle obtenue au sous-paragraphe א(1). Montrez la trace.

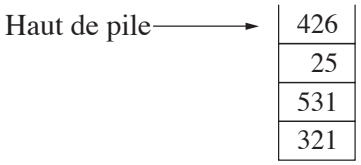
(3) Écrivez en une phrase, ce qu'exécutera la méthode booléenne Secret1 , c'est à dire à quelle question la méthode retourne true ou false .

/suite en page 7/

Soit la méthode Secret2 :

```
public static bool Secret2 ( Stack <int> s )
    {
        bool ok;
        int x;
        if ( s.IsEmpty () )
            ok = true ;
        else
        {
            x = s.Pop () ;
            if ( ! (Secret1 (x, x % 10)) )
                ok = false;
            else
                ok = Secret2 (s);
        }
        return ok ;
    }
```

ב. **(1)** Pour la pile s ci-dessous :

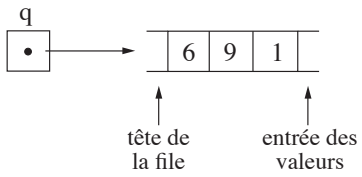


Écrivez ce que retournera la méthode Secret2 . Montrez la trace (il n'est pas nécessaire de montrez la trace de la méthode Secret1).

(2) Écrivez en une phrase, ce qu'exécutera la méthode booléenne Secret2 , c'est à dire à quelle question la méthode retourne true ou false .

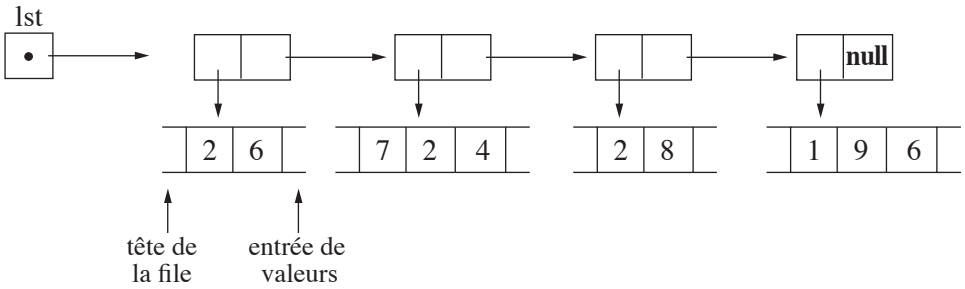
/suite en page 8/

5. **א.** Une "file nombre" est une file de chiffres de 1 à 9 (inclus), représentant un nombre entier – le premier élément (en tête de la file) est le chiffre des unités, le deuxième élément est le chiffre des dizaines et ainsi de suite.
Supposez que le nombre de chiffres possibles dans la file n'est pas supérieur au nombre de chiffres que peut contenir la variable de type int .
Par exemple : la file ci-dessous représente le nombre 196.



Écrivez une méthode toNumber en Java ou ToNumber en C#, acceptant une "file nombre" – q et retournant le nombre représenté par la file.
Remarque : il n'est pas nécessaire de conserver la structure de la file.

ב. Soit une liste chaînée dans laquelle chaque chaîne comprend une "file nombre".
Par exemple : la liste chaînée ci-dessous représente les nombres 62 , 427 , 82 , 691 .



Écrivez une méthode bigNumber en Java ou BigNumber en C#, acceptant une référence lst à la liste chaînée, et retournant le plus grand nombre représenté dans la liste chaînée.
Pour la liste chaînée décrite dans l'exemple, la méthode retournera le nombre 691 .
Il est obligatoire d'utiliser la méthode définie au paragraphe א.

/suite en page 9/

6. Soit la classe **Range** comportant deux attributs :

low – nombre de type entier.

high – nombre de type entier.

high est plus grand que low .

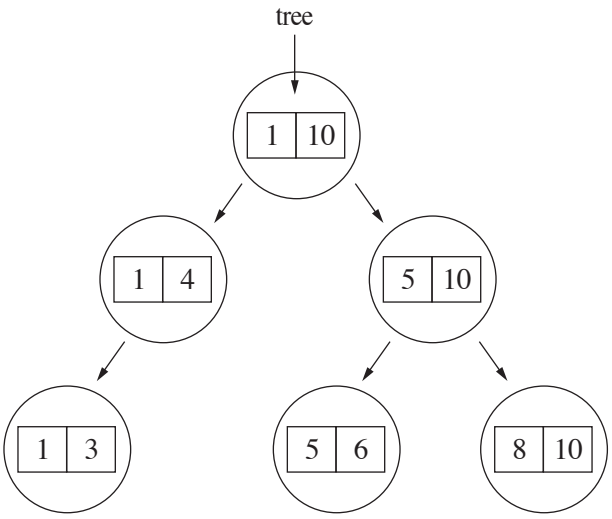
Supposez que pour chaque attribut, les méthodes get et set en Java et les méthodes Get et Set en C#, ont été définies.

Un arbre intervalle est un arbre dont les valeurs sont de type **Range** .

Un arbre intervalle ordonné est un arbre vide ou un arbre intervalle dans lequel pour chaque nœud, les conditions suivantes sont vérifiées :

- S'il existe un fils gauche, alors le low du nœud est égal au low du fils gauche, et le high du nœud est supérieur ou égal au high du fils gauche.
- S'il existe un fils droit, alors le high du nœud est égal au high du fils droit, et le low du nœud est inférieur ou égal au low du fils droit.
- S'il existe deux fils, alors le high du fils gauche est inférieur au low du fils droit.

Exemple d'un **arbre intervalle ordonné** :



Écrivez une méthode externe booléenne order en Java ou Order en C# qui accepte un **arbre intervalle** ou un **arbre vide**, et retourne true si l'arbre est un **arbre intervalle ordonné**, autrement – la méthode retourne false .

/suite en page 10/

Troisième section (25 points)

Cette section comporte des questions concernant quatre filières :
מערכות מחשב ואסמבלי, pages 10-11.
מבוא לחקר ביצועים, pages 12-15.
מודלים חישוביים, page 16.
תכנות מונחה עצמים ב–Java, pages 17-20 ; תכנות מונחה עצמים ב–C#, pages 21-24.
Répondez à une question de la filière dans laquelle vous avez étudié.

מערכות מחשב ואסמבלי

Si vous avez étudié dans cette filière, répondez à l'une des questions 7-8 (25 points).

7. Cette question comporte quatre paragraphes, א-ד, qui ne sont pas liés. Répondez à tous les paragraphes.

א. Voici un fragment de programme en assembleur.

```
        MOV BX, 100H
        MOV SI, 2
        MOV CX, 3
L1:
        MOV AL, CL
        MOV [BX], AL
        INC BX
        LOOP L1
L2:
        DEC AL
        MOV [BX], AL
        INC BX
        DEC SI
        JNZ L2
        NOP
```

À l'aide d'un tableau de trace, tracez l'exécution du fragment de programme. Insérez dans le tableau une colonne pour chacun des registres représentés dans le fragment. De plus, tracez un plan de mémoire correspondant.

/suite en page 11/

ב. Voici un fragment de programme écrit dans les langages Java et C#. Écrivez un fragment correspondant en assembleur.
Supposez que les variables a , b , c sont stockées respectivement dans les registres AX , BX , et CX .
Supposez que les nombres sont signés (Signed).

ג. Voici deux fragments de programme – le fragment 1 et le fragment 2.
À la fin de l'exécution de chacun des deux fragments, les valeurs dans le registre CX seront-elle égales ? Justifiez.

fragment 2	**fragment 1**
MOV CX, 10	MOV AX, 10
SHL CX, 1	MOV CX, AX
MOV DX, CX	MUL CX
SHL CX, 2	MOV CX, AX
ADD CX, DX	

ד. Le registre AX comporte un nombre quelconque. Écrivez des instructions inversant le byte inférieur et le byte supérieur du registre.
Par exemple : si la valeur initiale de AX est 5A9Ch , après l'inversement, la valeur sera 9C5Ah .
N'utilisez pas les fonctions ROR , ROL .

8. Dans un tableau, un "nombre fréquent" est le nombre qui apparaît le plus de fois dans le tableau. Par exemple, dans le tableau ci-dessous :

index:	0	1	2	3	4	5	6	7	8	9
valeur	− 3	6	0	11	8	− 9	5	6	33	6

Le "nombre fréquent" est 6 car le nombre de ses apparitions dans le tableau est 3, et il n'y a pas d'autre nombre qui apparaît dans le tableau plus de 3 fois.

א. Soit le tableau ARR de longueur 100 comportant des nombres de type byte.
Écrivez une procédure acceptant à travers la pile, un nombre entier, et stockant dans le registre AL , le nombre de fois que le nombre apparaît dans le tableau ARR .

ב. Écrivez un programme qui trouve le "nombre fréquent" dans le tableau ARR et le stocke dans la variable VAL .
Utilisez la procédure que vous avez écrite au paragraphe א.
Supposez que le tableau possède un "nombre fréquent" unique.

/suite en page 12/

מבוא לחקר ביצועים

Si vous avez étudié dans cette filière, répondez à l'une des questions 9-10 (25 points).

9. Soit le problème de programmation linéaire :

$\min\{z = 3x_1 + 5x_2\}$

obéissant aux contraintes suivantes :

(1) $x_1 + x_2 \geq 5$

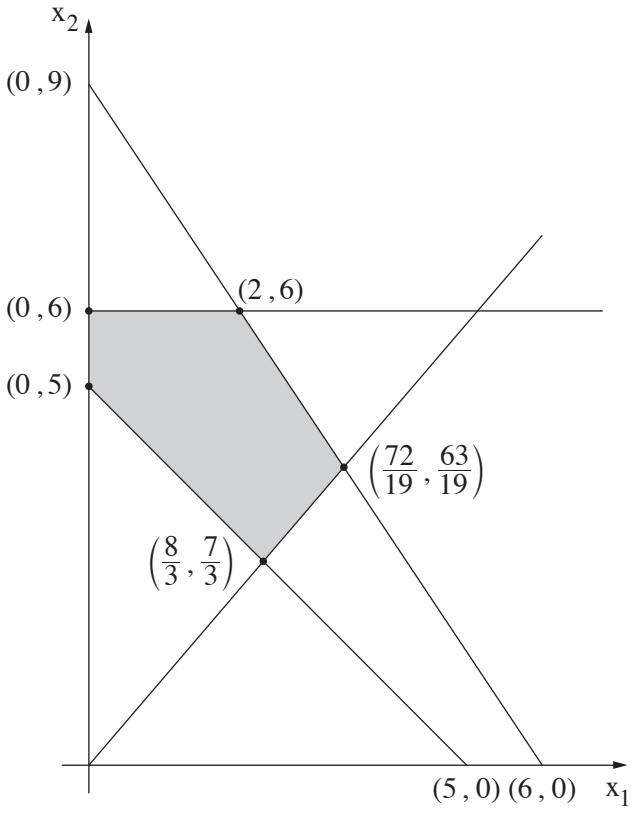
(2) $x_1 \geq 0$

(3) $0 \leq x_2 \leq 6$

(4) $7x_1 \leq 8x_2$

(5) $3x_1 + 2x_2 \leq 18$

Voici le diagramme des solutions admissibles au problème donné.



/suite en page 13/

Chacun des paragraphes א-ד suivants se rapporte au problème de programmation linéaire donné.

Les paragraphes א-ד ne sont pas liés. Répondez à tous les paragraphes.

Quatre affirmations i-iv sont proposées. Pour chacun des paragraphes א-ד ci-dessous, une seule affirmation est vraie.

i Il existe une unique solution optimale.

ii Il existe une infinité de solutions optimales.

iii La solution optimale n'est pas bornée.

iv Il n'existe pas de solution optimale.

Pour chacun des paragraphes א-ד ci-dessous, déterminez laquelle des affirmations i-iv est la vraie. Mentionnez dans votre cahier le numéro du paragraphe (א-ד), copiez-y l'affirmation correcte, puis justifiez votre réponse.

– Si vous avez établi pour un article quelconque, que l'affirmation i est la vraie – trouvez la solution optimale unique de ce paragraphe, ainsi que la valeur de la fonction objectif de cette solution.

– Si vous avez établi pour un article quelconque, que l'affirmation ii est la vraie – trouvez la solution optimale générale du problème, ainsi que la valeur de la fonction objectif dans l'intervalle des solutions optimales.

א. Quelle affirmation est vraie pour le problème de programmation linéaire donné au début de la question ? Justifiez votre réponse.

ב. On remplace uniquement la fonction objectif du problème donné au début de la question par $\max\{z = 3x_1 + 5x_2\}$.
Quelle affirmation est vraie suite à cette modification ? Justifiez votre réponse.

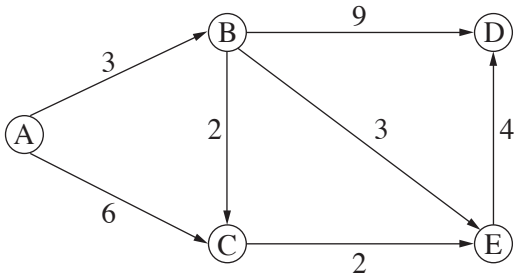
ג. On remplace uniquement la fonction objectif du problème donné au début de la question par $\min\{z = 5x_1 + 5x_2\}$.
Quelle affirmation est vraie suite à cette modification ? Justifiez votre réponse.

ד. On ajoute une contrainte au problème donné au début de la question : $x_1 > 2x_2$.
Quelle affirmation est vraie suite à l'ajout de cette contrainte ? Justifiez votre réponse.

/suite en page 14/

10. Cette question comporte deux paragraphes, א-ב, qui ne sont pas liés. Répondez aux deux.

א. G = (V , E) est un **graphe orienté** :



(1) Représentez le graphe à l'aide d'une matrice d'adjacence.

(2) Trouvez les parcours les plus courts entre le sommet A et chacun des nœuds du graphe. Décrivez chacun des parcours de manière schématique.

ב. **(1)** Dans le tableau ci-dessous, on donne une solution de base admissible obtenue après k itérations pour un problème de transport, et on donne la valeur de u_2 .

Origines	**Destinations**				**Offre**	u_i
	A	B	C	D		
1	[8] (30)	[7]	[3]	[2]	30	
2	[4] (10)	[9] (70)	[4] (10)	[4]	90	0
3	[3]	[2]	[7] (40)	[8] (40)	80	
Demande	40	70	50	40		
v_j						

i. Recopiez le tableau dans votre cahier, puis complétez-y les valeurs $u_1 , u_3 , v_1 , v_2 , v_3 , v_4$.

ii. Expliquez pourquoi la solution donnée n'est pas optimale.

iii. Exécutez une itération supplémentaire, c'est à dire $k + 1$. Quelle est la variable qui sort de la base dans cette itération ?

iv. Tracez dans votre cahier un nouveau tableau, puis inscrivez-y la solution qui sera obtenue suite à cette itération.

v. Le coût total du problème de transport a-t-il baissé ?

/suite en page 15/

(2) Le tableau ci-dessous représente un problème de transport.

Origines	**Destinations**			**Offre**
	A	B	C	
1	10	5	7	10
2	12	8	4	18
3	8	2	5	10
Demande	16	12	10	

i. On change la demande de la destination A de 16 en 10 , sans changer les autres demandes et les autres offres du tableau.
Tracez dans votre cahier un nouveau tableau comprenant toutes les données du tableau ci-dessus, et qui permettra de trouver une solution de base admissible, selon la méthode du coin nord-ouest. Insérez dans le tableau les prix de chaque destination.

ii. Déterminez la solution de base admissible selon la méthode du coin nord-ouest.

/suite en page 16/

מודלים חישוביים

Si vous avez étudié dans cette filière, répondez à l'une des questions 11-12 (25 points).

11. Voici les langages L_1 et L_2 sur l'alphabet $\{0,1\}$:

$L_1 = \{0^i 1^{2n} \mid i \geq 1,\ n = i \bmod 3\}$

$L_2 = \{0^i 1^{n+i} \mid i \geq 1,\ n = i \bmod 3\}$

א. Écrivez un mot d'une longueur de 6 appartenant au langage L_1 et un mot d'une longueur de 6 appartenant au langage L_2 .

ב. Si le langage L_1 est régulier – construisez un automate fini déterministe non complet qui acceptera le langage, et si le langage n'est pas régulier – construisez un automate pile qui acceptera ce langage.

ג. Si le langage L_2 est régulier – construisez un automate fini déterministe non complet qui acceptera ce langage, et si le langage n'est pas régulier – construisez un automate pile qui acceptera ce langage.

12. Construisez une machine de Turing acceptant comme entrée des mots sur l'alphabet $\{1,0\}$.
La machine retournera le mot obtenu sans les zéros, c'est à dire que le mot qui sera retourné comprendra uniquement les chiffres 1, entre deux signes $ à un endroit quelconque du ruban.
Par exemple, si le ruban se présente ainsi :

⊢	1	1	0	0	1	0	1	△	△	....

La sortie peut par exemple ressembler à cela :

....	△	$	1	1	1	1	$	△	.....

/suite en page 17/

תכנות מונחה עצמים en Java

Si vous avez étudié dans cette filière et que vous rédigez en Java, répondez à l'une des questions 13-14. (25 points).

13. Soit les classes : A , B , Driver

```
public class A {
    private int val;
    public A () {
        this.val = 1;
    }
    public A (int val) {
        this.val = val;
    }
    public int getVal () {
        return this.val;
    }
    public boolean equals (Object other) {
        System.out.println ("AObject");
        if (other instanceof A)
            return (this.val == ((A) other).val);
        return false;
    }
}
```

```
public class B extends A {
    private String st;
    public B () {
        this.st = "B" ;
    }
    public B (String st, int val) {
        super (val);
        this.st = st;
    }
    public String getSt () {
        return this.st;
    }
    public boolean equals (Object other) {
        System.out.println ("BObject") ;
        if (other instanceof B)
            return (this.st.equals (((B) other).st)
            && this.getVal () == ((B) other).getVal ()) ;
        return false;
    }
    public boolean equals(A other) {
        System.out.println ("BA") ;
        if (other instanceof B)
            return (this.st.equals (((B) other).st)
            && this.getVal () == ((B) other).getVal ()) ;
        return false;
    }
    public boolean equals (B other) {
        System.out.println ("BB") ;
        return (this.st.equals (other.st)
        && this.getVal () == other.getVal ()) ;
    }
}
```

/suite en page 18/

```
public class Driver {
    public static void main (String[] args) {
        A a1 = new A () ;
        A a2 = new A (5) ;
        A ab = new B () ;
        B b1 = new B ("B", 1) ;
        B b2 = new B ("B", 5) ;
        //  *** //
    }
}
```

א. Dessinez les objets créés par la méthode main . Mentionnez pour chaque objet, les valeurs de ses attributs.

ב. Voici huit instructions, 1-8 :

- Substituez chacune de ces instructions à la place de //***// dans la méthode main de la classe Driver .
- Écrivez ce que sera la sortie suite à la substitution de chacune de ces instructions.

1. if (a1.equals (b1)) System.out.println (1) ;
2. if (b1.equals (a1)) System.out.println (2) ;
3. if (a1.equals (ab)) System.out.println (3) ;
4. if (ab.equals (a1)) System.out.println (4) ;
5. if (b1.equals (ab)) System.out.println (5) ;
6. if (ab.equals (b1)) System.out.println (6) ;
7. if (a1.equals (a2)) System.out.println (7) ;
8. if (b1.equals (b2)) System.out.println (8) ;

/suite en page 19/

14. Soit les classes : Driver , First , Second , Third .

```
public class First {
    public static int count = 0 ;
    private String str ;

    public First (String str)  {
        count ++ ;
        this.str = str  ;
    }

    public First (First g)  {
        count ++ ;
        this.str = "Copy" + g.str ;
    }

    public void setStr (String str)   {
        this.str = str ;
    }

    public void print ()    {
        System.out.println ("First" + this.str) ;
    }
}
```

```
public class Second extends First  {
    private First f;
    public Second (First fx , First fy)   {
        super (fx) ;
        this.f = fy  ;
    }

    public void setStr (String str)   {
        super.setStr (str) ;
        this.f = new First (str) ;
    }

    public void print ()    {
        System.out.println ("Second") ;
        super.print () ;
        this.f.print () ;
    }
}
```

/suite en page 20/

```
public class Third  {
    private int curr;
    private First[] arr ;
    public Third (int count )  {
        curr = 0 ;
        arr = new First [count] ;
    }

    public void print ()  {
        for ( int i = 0 ; i < curr ; i++ )
            arr [i].print () ;
    }

    public void add ( First s )  {
        if ( curr < arr.length ) {
            arr [curr] = s ;
            curr ++ ;  }
    }
}
```

```
public class Driver  {
    public static void main (String[] args) {
        First f1 = new First ("One") ;
        First f2 = new First ("Two") ;
        First f3 = new First (f2) ;
        Second s1 = new Second (f1, f3) ;
        Third t = new Third (First.count) ;
        t.add (f1) ;
        t.add (f2) ;
        t.add (s1) ;
        System.out.println (First.count) ;
        t.print () ;
        System.out.println (" ----------") ;
        f1.setStr ("Five") ;
        System.out.println (First.count) ;
        t.print () ;
    }
}
```

א. Dessinez un diagramme de hiérarchie des classes First , Second , Third . Notez l'héritage à l'aide de la flèche ——▷ et la composition à l'aide du signe ——◆ .

ב. Tracez l'exécution de la méthode main de la classe Driver .

(1) Exposez le contenu de tous les objets créés.

(2) Écrivez la sortie obtenue.

/suite en page 21/

תכנות מונחה עצמים en C#

Si vous avez étudié dans cette filière et que vous rédigez en C#, répondez à l'une des questions 15-16. (25 points).

15. Soit les classes : A , B , Driver

```
public class A {
    private int val;

    public A () {
        this.val = 1;
    }

    public A (int val) {
        this.val = val;
    }

    public int GetVal () {
        return this.val;
    }

    public virtual bool Equals (Object other) {
        Console.WriteLine ("AObject") ;
        if (other is A)
            return (this.val == ((A) other).val) ;
        return false ;
    }
}
```

```
public class B : A {
    private string st ;

    public B () {
        this.st = "B" ;
    }
    public B (string st, int val)
        : base (val) {
        this.st = st ;
    }
    public string GetSt () {
        return this.st ;
    }
    public override bool Equals (Object other) {
        Console.WriteLine ("BObject ") ;
        if (other is B)
            return (this.st.Equals (((B) other).st)
            && this.GetVal () == ((B) other).GetVal ());
        return false ;
    }
    public bool Equals (A other) {
        Console.WriteLine ("BA");
        if (other is B)
            return (this.st.Equals (((B) other). st)
            && this.GetVal () == ((B) other).GetVal ()) ;
        return false;
    }
    public bool Equals (B other) {
        Console.WriteLine("BB");
        return (this.st.Equals (other. st)
        && this.GetVal () == other.GetVal ()) ;
    }
}
```

/suite en page 22/

```
public class Driver {
    public static void Main (string[] args) {
        A a1 = new A ();
        A a2 = new A (5);
        A ab = new B ();
        B b1 = new B ("B", 1);
        B b2 = new B ("B", 5);
        //  *** //
    }
}
```

א. Dessinez les objets créés par la méthode `Main` . Mentionnez pour chaque objet, les valeurs de ses attributs.

ב. Voici huit instructions, 1-8 :

- Substituez chacune des instructions à la place de `//***//` dans la méthode `Main` de la classe `Driver` .
- Écrivez ce que sera la sortie suite à la substitution de chacune de ces instructions.

```
1. if (a1.Equals (b1)) Console.WriteLine (1) ;
2. if (b1.Equals (a1)) Console.WriteLine (2) ;
3. if (a1.Equals (ab)) Console.WriteLine (3) ;
4. if (ab.Equals (a1)) Console.WriteLine (4) ;
5. if (b1.Equals (ab)) Console.WriteLine (5) ;
6. if (ab.Equals (b1)) Console.WriteLine (6) ;
7. if (a1.Equals (a2)) Console.WriteLine (7) ;
8. if (b1.Equals (b2)) Console.WriteLine (8) ;
```

/suite en page 23/

16. Soit les classes : Driver , First , Second , Third .

```
public class First
{
    public static int count=0 ;
    private string str ;

    public First (string str)  {
        count ++  ;
        this.str = str ;
    }

    public First (First g)  {
        count ++ ;
        this.str = "Copy" + g.str ;
    }

    public virtual void SetStr (string str)   {
        this.str = str ;
    }

    public virtual void Print ()    {
        Console.WriteLine ("First" + this.str ) ;
    }
}
```

```
public class Second : First  {
    private First f;

    public Second (First fx, First fy)
    : base (fx)   {
        this.f = fy ;
    }

    public override void SetStr (string str)   {
        base.SetStr (str) ;
        this.f = new First (str) ;
    }

    public override void Print ()    {
        Console.WriteLine ("Second") ;
        base.Print () ;
        this.f.Print ();
    }
}
```

/suite en page 24/

```
public class Third   {
        private int curr;
        private First[] arr ;

       public Third (int count)  {
             curr = 0 ;
             arr = new First [count] ;
       }

       public void Print ()   {
             for (int i = 0 ; i < curr ; i++)
                   arr [i].Print () ;
         }

       public void Add (First s)  {
             if (curr < arr.Length) {
                   arr [curr] = s ;
                   curr++ ;   }
       }
}
```

```
public class Driver
{
      public static void Main (string[] args) {
            First f1 = new First ("One") ;
            First f2 = new First ("Two") ;
            First f3 = new First (f2) ;
            Second s1 = new Second (f1, f3) ;
            Third t = new Third (First.count) ;
            t.Add (f1) ;
            t.Add (f2) ;
            t.Add (s1) ;
            Console.WriteLine (First.count) ;
            t.Print() ;
            Console.WriteLine (" ----------") ;
            f1.SetStr ("Five") ;
            Console.WriteLine (First.count) ;
            t.Print () ;
      }
}
```

א. Dessinez un diagramme de hiérarchie des classes First , Second , Third . Notez l'héritage à l'aide de la flèche ——▷ et la composition à l'aide du signe ——◆ .

ב. Tracez l'exécution de la méthode Main de la classe Driver .

(1) Exposez le contenu de tous les objets créés.

(2) Écrivez la sortie obtenue.

Bonne chance!

בהצלחה!