État d'Israël
Ministère de l'Éducation
Type d'examen : *Bagrout*
Session de l'examen : été 2018
Numéro de l'examen : 899381
Traduction française (4)

Informatique

Instructions au candidat

א. Durée de l'examen : trois heures.

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

Première section — Cette section comporte trois questions, répondez d'après les instructions énoncées dans cette section.
$(1 \times 10) + (1 \times 15)$ — 25 points

Deuxième section — Cette section comporte trois questions, répondez à deux d'entre elles.
(2×25) — 50 points

Troisième section — Cette section comporte des questions de quatre filières différentes. Répondez à une question de la filière dans laquelle vous avez étudié.
(1×25) — 25 points

Total — 100 points

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

ד. Instructions particulières :
(1) Tous les programmes qu'il vous est demandé d'écrire en langage informatique dans la première 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 :
עצמים, מודלים חישוביים, מבוא לחקר ביצועים, מערכות מחשב ואסמבלי תכנות מונחה.

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.

Tout ce que vous voulez écrire en tant que brouillon (plans, calculs, etc...), doit être écrit dans le cahier d'examen uniquement, sur des pages séparées. Indiquez "טיוטה" en haut de chaque page de brouillon. Écrire des notes de brouillon sur d'autres feuilles que celles du cahier d'examen peut entraîner votre disqualification !

Bonne chance!

מדינת ישראל
משרד החינוך
סוג הבחינה: בגרות
מועד הבחינה: קיץ תשע"ח, 2018
מספר השאלון: 899381
תרגום לצרפתית (4)

מדעי המחשב

הוראות לנבחן

א. משך הבחינה: שלוש שעות.

ב. מבנה השאלון ומפתח ההערכה:
בשאלון זה שלושה פרקים.
פרק ראשון – בפרק זה שלוש שאלות, ענה על פי ההוראות בפרק.
(1×10) + (1×15) — 25 נקודות

פרק שני – בפרק זה יש שלוש שאלות, ומהן עליך לענות על שתיים.
(2×25) — 50 נקודות

פרק שלישי – בפרק זה שאלות בארבעה מסלולים שונים.
ענה על שאלה אחת במסלול שלמדת.
(1×25) — 25 נקודות
סה"כ — 100 נקודות

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

ד. הוראות מיוחדות:
(1) את כל התכניות שאתה נדרש לכתוב בשפת מחשב בפרק הראשון 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.

כתוב במחברת הבחינה בלבד, בעמודים נפרדים, כל מה שברצונך לכתוב כטיוטה (ראשי פרקים, חישובים וכדומה). רשום "טיוטה" בראש כל עמוד טיוטה. רישום טיוטות כלשהן על דפים מחוץ למחברת הבחינה עלול לגרום לפסילת הבחינה!

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. Soit la classe **AllNumbers** comportant un attribut : un tableau unidimensionnel – arrayNum de type entier.
Ce tableau contient des nombres positifs supérieurs à zéro ; certains sont des nombres pairs et certains sont des nombres impairs.
Écrivez une méthode interne lastOddValue en Java ou LastOddValue en C# qui retournera la valeur du dernier nombre impair du tableau.
Par exemple : Pour le tableau arrayNum de taille 6 ci-dessous, la méthode retournera le nombre 3, qui correspond à la valeur du dernier nombre impair du tableau.

	0	1	2	3	4	5
arrayNum	7	5	8	9	**3**	4

Supposez que ce tableau comporte au moins un chiffre impair.

/suite en page 3/

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

2. 40 chansons numérotées de 1 à 40, participent au concours de la chanson de l'année.
25 juges classent ces chansons.
Chaque juge choisi les trois meilleures chansons selon lui. Il attribue 7 points à la chanson placée en première position (la meilleure), 5 points à la chanson placée en deuxième position et 1 point à la chanson placée en troisième position.
La chanson qui totalisera le plus de points chez l'ensemble des juges remportera la première place. À l'approche du concours de la chanson de l'année, on a défini une classe **Vote** .
Cette classe comporte trois attributs :

first – numéro de la chanson que le juge a placé en première position, de type entier.
second – numéro de la chanson que le juge a placé en deuxième position, de type entier.
third – numéro de la chanson que le juge a placé en troisième position, de type entier.

Supposez que pour chaque attribut, les méthodes get et set en Java et les méthodes Get et Set en C#, ont été définies.
Écrivez une méthode externe theWinner en Java ou TheWinner en C#, recevant un tableau de type **Vote** du classement des juges, et qui affiche le numéro de la chanson qui obtient la première position.
Supposez qu'une seule chanson gagne le concours.
Supposez que les valeurs du tableau sont valides.

/suite en page 4/

3. Soit la classe **Time** comportant deux attributs :
hour – représente une heure entre 0 et 23 inclus, de type entier.
minute – représente une minute entre 0 et 59 inclus, de type entier.
Supposez que pour chaque attribut, les méthodes get et set en Java et les méthodes Get et Set en C#, ont été définies.

א. Écrivez en Java ou en C# dans la classe **Time** une méthode constructeur acceptant des valeurs pour **tous** les attributs.
Pour chaque attribut – si la valeur obtenue est impossible d'après les définitions de cet attribut, elle sera remplacée par 0.

ב. Soit la classe **Flight** comportant quatre attributs : le nom de la compagnie aérienne – name de type chaîne, le nom du pays de la destination – destination de type chaîne, le code du vol – flightCode de type chaîne, et le temps de vol – flightTime de type **Time** .
Écrivez en Java ou en C# la déclaration de la classe et les attributs de la classe.

ג. Soit la classe **Airport** comportant un attribut : un tableau unidimensionnel – fligths de type **Flight** .
Supposez que pour chaque attribut de la classe **Flight** les méthodes get et set en Java et les méthodes Get et Set en C#, ont été définies.
Soit la classe **Airport** comportant, une méthode interne booléenne isFly en Java ou IsFly en C# , dont le but est de vérifier si le tableau des vols flights comporte un vol de la compagnie aérienne "sky" .
La méthode retournera true si le tableau flights comporte un vol de la compagnie aérienne "Sky". Autrement – la méthode retournera false .
Supposez que toutes les cellules du tableau sont différentes de null .

La méthode ci-dessous (isFly en Java ou IsFly en C#) est **erronée**.

Java

```
public boolean isFly()
{
    for (int i = 0; i < this.flights.length; i++)
    {
        if (this.flights[i].getName().equals ("Sky"))
            return true;
        return false;
    }
}
```

C#

```
public bool IsFly()
{
    for (int i = 0; i < this.flights.Length; i++)
    {
        if (this.flights[i].GetName() == "Sky")
            return true;
        return false;
    }
}
```

/suite en page 5/

(1) Dites ce que retournera la méthode **erronée** pour le tableau flights de taille 4, dont les valeurs de l'attribut name des élément du tableau sont :

en première position : "Cloud" ,
en deuxième position : "Air" ,
en troisième position : "Sky" ,
en quatrième position : "Travel" .

(2) Expliquez quelle est l'erreur dans la méthode.

(3) Corrigez la méthode afin qu'elle effectue ce qui est requis, puis recopiez dans votre cahier la méthode corrigée.

/suite en page 6/

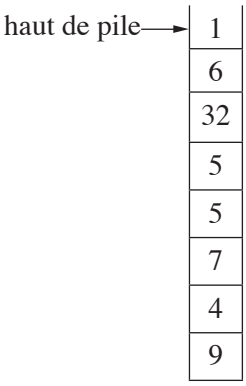
Deuxième section (50 points)

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.

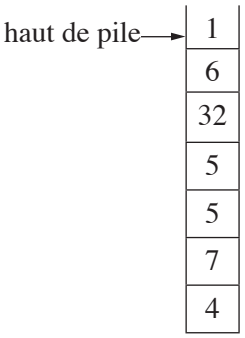
Répondez à deux des questions 4-6 (25 points par question).

4. א. Écrivez une méthode externe lastAndRemove en Java ou LastAndRemove en C#, qui reçoit une pile de type entier, efface l'élément inférieur de la pile et retourne sa valeur. Au terme de la méthode, les autres éléments de la pile restent inchangés.
Supposez que la pile n'est pas vide.

Par exemple, pour la pile ci-dessous :



au terme de la méthode, la valeur 9 sera retournée et la pile ressemblera à cela :



/suite en page 7/

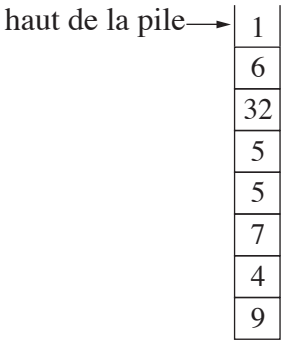
ב. Soit la classe :

TwoItems	
attribut :	int num1 int num2
méthode constructeur	TwoItems (int number1, int number2)

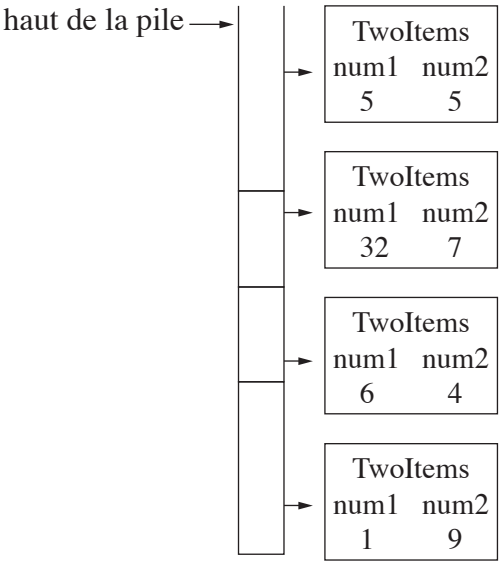
Supposez que pour chaque attribut les méthodes get et set en Java et les méthodes Get et Set en C#, ont été définies.
Écrivez une méthode externe stackTwoItems en Java ou StackTwoItems en C#, qui reçoit une pile stk1 qui n'est pas vide, de type entier et de taille paire, et retourne une pile de type TwoItems .

L'élément situé en bas de la pile retournée comprendra un pointeur vers l'instance TwoItems dont l'attribut num1 est l'élément situé en haut de la pile stk1 , et son attribut num 2 est l'élément situé en bas de la pile stk1 .
L'élément se trouvant au dessus de l'élément situé en bas de la pile retournée comprendra un pointeur vers l'instance TwoItems dont l'attribut num1 est un élément qui se trouve sous l'élément situé en haut de la pile stk1 , et son attribut num 2 est un élément qui se trouve au dessus de l'élément situé en bas de la pile stk1 , et ainsi de suite de sorte que l'élément se trouvant en haut de la pile retournée comprendra un pointeur vers l'instance TwoItems dont les attributs num1 et num 2 sont deux éléments consécutifs situés au milieu de la pile stk1 .

Par exemple,
pour la pile stk1 ci-dessous :



au terme de la méthode,
la pile retournera :



Aidez-vous de la méthode que vous avez écrite au paragraphe א .
Remarque : il n'est pas nécessaire de conserver le contenu original de la pile stk1 .

/suite en page 8/

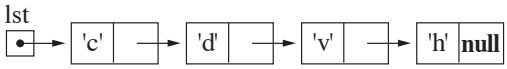
5. <u>Attention : Cette question comporte une version en Java en page 8, et une autre version en C# en page 9</u>.

<u>Pour ceux qui résolvent en Java</u> :

Soit la méthode sod1 :

```
public static Node <Character> sod1(Node <Character> lst, char ch)
{
    if (lst == null)
        return null;
    if (lst.getValue() == ch)
        return lst;
    return sod1(lst.getNext(), ch);
}
```

א. **(1)** Tracez l'exécution de la méthode, puis donnez ce qui sera retourné pour ch = 'v' et pour le pointeur lst vers une liste chaînée de caractères :



(2) Quelle est le but de la méthode sod1 ?

(3) Quelle est la complexité d'exécution de la méthode sod1 ? Justifiez.

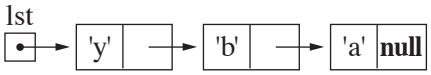
ב. Soit la méthode sod2 :

```
public static boolean sod2(Node <Character> lst)
{
    if (sod1(lst,'a') != null && sod1(lst,'b') != null)
        return true;
    return false;
}
```

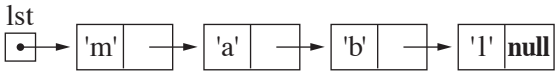
Quelle est le but de la méthode sod2 ?

ג. Écrivez une méthode booléenne recevant un pointeur vers une liste chaînée de caractères qui retournera true si dans cette méthode deux maillons de valeurs 'a' 'b' <u>ou</u> 'b' 'a' apparaissent de façon consécutive. Autrement – la méthode retournera false .

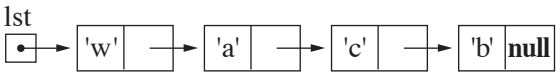
Exemple d'une liste chaînée où 'b' 'a' apparaissent de façon consécutive :



Exemple d'une liste chaînée où 'a' 'b' apparaissent de façon consécutive :



Exemple d'une liste chaînée où 'a' 'b' et b' 'a' n'apparaissent pas de façon consécutive :



Utilisez la méthode sod1 .

Supposez que tous les caractères sont différents les uns des autres.

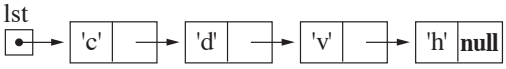
/suite en page 9/

Pour ceux qui résolvent C# :

Soit la méthode Sod1 :

```
public static Node <char> Sod1(Node <char> lst, char ch)
{
    if (lst == null)
        return null;
    if (lst.GetValue() == ch)
        return lst;
    return Sod1(lst.GetNext(), ch);
}
```

א. **(1)** Tracez l'exécution de la méthode, puis donnez ce qui sera retourné pour ch = 'v' et pour le pointeur lst vers une liste chaînée de caractères :



(2) Quelle est le but de la méthode Sod1 ?

(3) Quelle est la complexité d'exécution de la méthode Sod1 ? Justifiez.

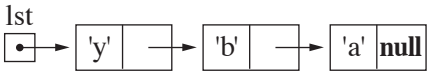
ב. Soit la méthode Sod2 :

```
public static bool Sod2(Node <char> lst)
{
    if (Sod1(lst,'a') != null && Sod1(lst,'b') != null)
        return true;
     return false;
}
```

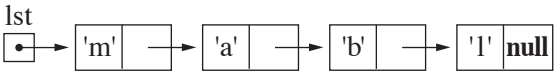
Quelle est le but de la méthode Sod2 ?

ג. Écrivez une méthode booléenne recevant un pointeur vers une liste chaînée de caractères qui retournera true si dans cette méthode deux maillons de valeurs 'a' 'b' ou 'b' 'a' apparaissent de façon consécutive. Autrement – la méthode retournera false .

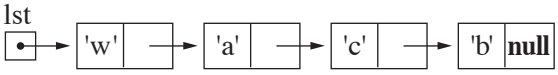
Exemple d'une liste chaînée où 'b' 'a' apparaissent de façon consécutive :



Exemple d'une liste chaînée où 'a' 'b' apparaissent de façon consécutive :



Exemple d'une liste chaînée où 'a' 'b' et b' 'a' n'apparaissent pas de façon consécutive :



Utilisez la méthode Sod1 .

Supposez que tous les caractères sont différents les uns des autres.

/suite en page 10/

6. Soit la méthode :

En Java :

public static boolean lessThanTree (BinNode <Integer> t, int x)

En C# :

public static bool LessThanTree (BinNode <int> t, int x)

La méthode retournera true si x est plus petit que les valeurs de l'arbre t . Autrement – la méthode retournera false .

La complexité de l'exécution de la méthode est O(n) . n représente le nombre de nœuds que contient l'arbre t .

א. Écrivez une méthode externe treeLessThanTree en Java ou TreeLessThanTree en C# qui accepte deux arbres binaires t1 et t2 dont les valeurs sont des nombres entiers.
Il est donné que t2 possède au moins un nœud. La méthode retournera true si chaque valeur de l'arbre t1 est plus petite que chacune des valeurs de l'arbre t2 , autrement – la méthode retournera false .
Si t1 est null , la méthode retournera true .
Vous pouvez utiliser la méthode donnée sans l'implémenter. Si vous utilisez d'autres méthodes, implémentez-les.

ב. Quelle est la complexité d'exécution de la méthode que vous avez écrite dans le paragraphe א ? Justifiez.

/suite en page 11/

Troisième section (25 points)

Cette section comporte des questions concernant quatre filières :

מערכות מחשב ואסמבלי, pages 11-13.

מבוא לחקר ביצועים, pages 14-17.

מודלים חישוביים, page 18-19.

תכנות מונחה עצמים ב–Java, pages 20-23 ; תכנות מונחה עצמים ב–C#, pages 24-27.

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 deux articles, א-ב, qui ne sont pas liés. Répondez aux deux.

א. Voici une portion de programme en assembleur.

Supposez que les nombres ne sont pas signés (Unsigned).

```
        CMP  AH , AL
        JAE  AA
        XCHG AH, AL
AA:     CMP AL, AH
        JAE CC
        INC AL
        DEC AH
        JMP AA
CC:     MOV SI, 100H
        MOV [SI], AL
```

(1) Le contenu du registre AX est 0109H . À l'aide d'un tableau de trace, tracez l'exécution de la portion de programme, puis écrivez ce qu'on obtiendra après l'exécution dans la case dont l'adresse est 100H.

(2) Qu'exécute la portion de programme pour des nombres impairs dans les registres AL , AH ?

Dans votre réponse, traitez également du contenu des registres AL , AH avant l'exécution de la portion de programme.

/suite en page 12/

ב. (Il n'y a pas de lien avec l'article א)
Voici six affirmations. Pour chacune d'entre-elles, déterminez si elle est vraie ou fausse.
Si l'affirmation est fausse, expliquez pourquoi.

(1) Les deux portions 1 et 2 ci-dessous exécutent la même chose.

portion 1	portion 2
SHL AX, 1	SHR AX, 1
SHR AX, 1	SHL AX, 1

(2) Les deux portions 1 et 2 ci-dessous exécutent la même chose.

portion 1	portion 2
AND AX, 0FEh	SHR AX, 1
	SHL AX, 1

(3) Les deux portions 1 et 2 ci-dessous exécutent la même chose.

portion 1	portion 2
PUSH AX	XCHG AX, BX
PUSH BX	
POP AX	
POP BX	

(4) Les deux portions 1 et 2 ci-dessous exécutent la même chose.

portion 1	portion 2
SUB AX, 0	CMP AX, 0
JNE A1	JNZ A1

(5) Dans le segment de données on a défini le tableau :

```
ARR    DW 50 dup (?)
```

Supposez que dans le registre BX on a stocké l'adresse d'un certain élément du tableau ARR .
Cette portion de programme stocke l'indice de ce même élément dans le registre BX .

```
LEA SI, ARR
SUB BX,SI
INC BX
```

(6) La valeur du nombre 11101011 qui est stockée selon la méthode : le complément à 2 en 8 bits est (-21) en base 10.

/suite en page 13/

8. Dans le segment de données ci-dessous, on a défini les données :

ARR DB 100 DUP (?)

REZ DB ?

Supposez que les nombres sont signés (Signed).
Supposez que tous les nombres dans le tableau sont différents les uns des autres.

Un tableau est dit "ondulatoire" s'il répond aux conditions suivantes :
Le premier élément est inférieur au deuxième élément, le deuxième élément est supérieur au troisième élément, le troisième élément est inférieur au quatrième élément, le quatrième élément est supérieur au cinquième élément, et ainsi de suite.
Exemple d'un tableau "ondulatoire" :

0	1	2	3	4
−2	8	6	17	−7

Explication de l'exemple : le premier élément (−2) est inférieur au deuxième élément (8), le deuxième élément (8) est supérieur au troisième élément (6), le troisième élément (6) est inférieur au quatrième élément (17), le quatrième élément (17) est supérieur au cinquième élément (−7).

Écrivez une portion de programme vérifiant si le tableau ARR est un tableau "ondulatoire".
Si oui, vous affecterez la valeur 1 à la variable REZ . Si non, vous affecterez la valeur 0 à la variable REZ .

/suite en page 14/

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 :

$$\max\{z = 4x_1 + 6x_2\}$$

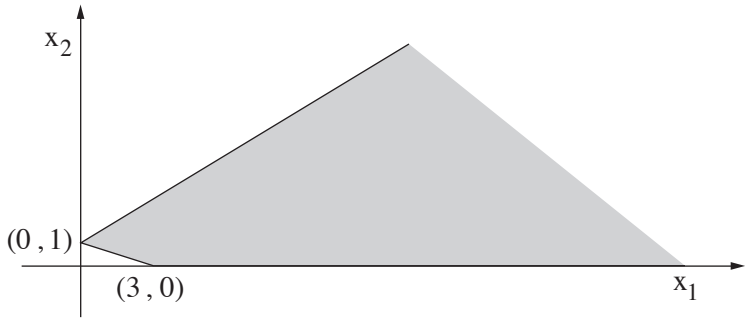
obéissant aux contraintes suivantes :

(1) $-5x_1 + 4x_2 \leq 4$

(2) $x_1 + 3x_2 \geq 3$

(3) $x_2 \geq 0$

Voici le diagramme des solutions admissibles au problème donné.



Chacun des articles א-ו de la page suivante se rapporte au problème de programmation linéaire donné.
Les articles א-ו ne sont pas liés. Répondez à tous les articles.
Quatre affirmations i-iv sont proposées. Pour chacun des articles א-ו de la page suivante, une seule affirmation est vraie.

i La solution optimale est unique.
ii Il existe une infinité de solutions optimales.
iii La solution optimale n'est pas bornée.
iv Il n'existe pas de solution optimale.

/suite en page 15/

Pour chacun des articles א-ו , déterminez laquelle des affirmations i-iv est la vraie.
Mentionnez l'article, copiez dans votre cahier 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 cet article, 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 programme donné au début de la question par $\min\{z = 4x_1 + 6x_2\}$.
Quelle affirmation est vraie suite à cette modification ? Justifiez votre réponse.

ג. On remplace uniquement la fonction objectif du programme donné au début de la question par $\max\{z = 2x_1 + 6x_2\}$.
Quelle affirmation est vraie suite à cette modification ? Justifiez votre réponse.

ד. On remplace uniquement la fonction objectif du programme donné au début de la question par $\min\{z = 2x_1 + 6x_2\}$.
Quelle affirmation est vraie suite à cette modification ? Justifiez votre réponse.

ה. On ajoute une contrainte au problème donné au début de la question : $x_1 + x_2 \leq 1$.
Quelle affirmation est vraie suite à l'ajout de cette contrainte ? Justifiez votre réponse.

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

/suite en page 16/

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

א. G = (V , E) est un graphe **orienté** représenté par la matrice d'adjacence ci-dessous :

	a	**b**	**c**	**d**	**e**
a	0	0	1	1	0
b	1	0	0	1	0
c	0	0	0	0	1
d	0	1	0	0	0
e	1	0	0	0	0

(1) Tracez le graphe G représenté par la matrice d'adjacence.

(2) Trouvez les composantes fortement connexes (Strong Connected Components – CFC) du graphe donné. Pour chaque CFC que vous avez trouvée, donnez le groupe de ses sommets.

(3) Quel est le nombre maximal d'arcs qu'il est possible de retirer du graphe donné, pour que celui-ci comporte toujours le même nombre de CFC que vous avez déterminée à l'article א (2) .
Quel est cet arc ou quels sont ces arcs ?

ב. (Il n'y a pas de lien avec l'article א)

(1) Dans le tableau ci-dessous, on donne une solution de base admissible partielle à un problème de transport : $x_{11} = 9$, $x_{12} = 1$.

Origines	**Destinations**			**Offre**
	1	2	3	
1	1 (9)	5 (1)	7	10
2	1	8	4	11
3	5	2	8	10
Demande	9	12	10	

Recopiez le tableau dans votre cahier, puis complétez-y les valeurs selon la méthode du coin nord-ouest.

/suite en page 17/

(2) Dans le tableau ci-dessous, on donne une solution de base admissible partielle à un problème de transport, ainsi que les valeurs de u_1 , u_2 , u_3 , v_1 , v_2 , v_3 .

Origines	**Destinations**			**Offre**	u_i
	1	2	3		
1	[3] (20)	[5]	[7]	20	1
2	[2]	[8] (10)	[14]	10	0
3	[2]	[6]	[8] (10)	15	−2
Demande	20	15	10		
v_j	2	8	10		

Recopiez le tableau dans votre cahier, puis complétez-le en tenant compte des valeurs de tous les u_i et de tous les v_i , afin d'obtenir une solution de base admissible.

(3) Dans le tableau ci-dessous, on donne une solution de base admissible à un problème de transport, et l'on donne les valeurs de u_1 , u_2 , u_3 , v_1 , v_2 , v_3 .

Origines	**Destinations**			**Offre**	u_i
	1	2	3		
1	[34]	[15] (15)	[17] (3)	18	0
2	[10] (10)	[8] (0)	[4]	10	−7
3	[25]	[18]	[18] (10)	10	1
Demande	10	15	13		
v_j	17	15	17		

La solution est-elle optimale ? Justifiez votre réponse.

/suite en page 18/

מודלים חישוביים

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

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

א. Voici une définition : l'**antécédent** d'un mot x est tout mot obtenu par la suppression d'un nombre quelconque de caractères depuis la fin du mot x , y compris le mot vide et le mot x lui-même.

Par exemple : pour le mot $x = abcbad$, tous les antécédents du mot x sont :

$\varepsilon, a, ab, abc, abcb, abcba, abcbad$

Voici le langage L sur l'alphabet $\Sigma = \{a,b,c,d\}$.

L est l'ensemble des mots pour lesquels dans chacun d'eux et pour **chaque antécédent** du mot, la différence entre le nombre de fois qu'apparaît le caractère c et le nombre de fois qu'apparaît le caractère d est supérieure ou égale à 0, et inférieure ou égale à 3 :

$0 \leq \#_c(w) - \#_d(w) \leq 3$

$\#_c(w)$ représente le nombre d'apparitions de c dans le mot w .

$\#_d(w)$ représente le nombre d'apparitions de d dans le mot w .

Exemples de mots appartenant au langage L :

accbdcacab , bacaabdbcb , abba , cdcdcd , abcbadb

Exemples de mots n'appartenant pas au langage L :

daac — car il existe un antécédent d pour lequel : $\#_c(w) - \#_d(w) = -1 < 0$

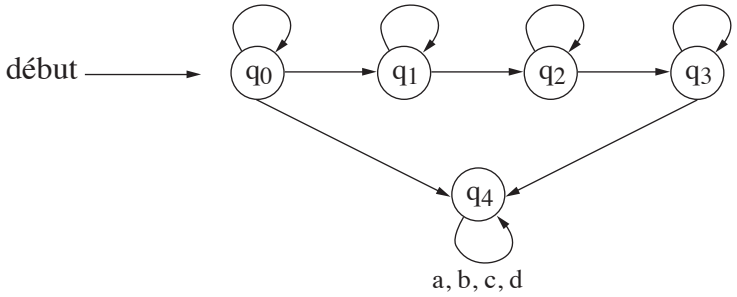
cddc — car il existe un antécédent cdd pour lequel : $\#_c(w) - \#_d(w) = -1 < 0$

accbdcacacd — car il existe un antécédent $accbdcacac$ pour lequel :

$\#_c(w) - \#_d(w) = 4 > 3$

/suite en page 19/

Voici le diagramme partiel d'un automate fini **déterministe** acceptant le langage L. Il manque à ce diagramme des transitions, des symboles d'entrée et des états acceptants. Le diagramme comporte tous les états de l'automate.



Recopiez ce diagramme dans votre cahier, puis complétez-le de manière à ce que l'automate soit **déterministe** et accepte le langage L.
Complétez les transitions manquantes et les symboles d'entrée manquants, puis signalez **tous les états acceptants.**
Attention : il ne faut pas ajouter d'états à l'automate, et il ne faut pas lui enlever ni d'états ni de transition.

ב. (Il n'y a pas de lien avec l'article א).

Σ^* est l'ensemble des mots sur l'alphabet Σ, en incluant le mot vide.

Soit les deux langages L_1, L_2 sur l'alphabet Σ.

$L_1 = \Sigma^*$ et L_2 est un langage qui n'est pas régulier.

On définit : $L_3 = L_2 \cap \overline{L_1}$.

(1) Qu'est-ce-que le langage $\overline{L_1}$?

(2) Le langage L_3 est-il régulier ? Justifiez votre réponse.

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

א. Soit l'alphabet $\Sigma = \{a,b\}$. Construisez un automate pile acceptant le langage :

$L = \{a^2 b^k a^n \mid n > 0 \quad , \quad k > 0 \quad , \quad n < k\}$

ב. (Il n'y a pas de lien avec l'article א).

Construisez une machine de Turing acceptant comme entrée un nombre x sous forme unaire et calculez la valeur de la fonction ci-dessous.

$$f(x) = \begin{cases} x+1 & x < 2 \\ x-1 & x \geq 2 \end{cases}$$

/suite en page 20/

תכנות מונחה עצמים

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

13. Voici un projet dans lequel ont été définies les classes suivantes : A , B , C , D , E . Dans ce projet la méthode f() a été implémentée dans deux classes.
La portion de code ci-dessous est **valide**.

```
A a1 = new A();
A e1 = new E();
E c1 = new C();
C b1 = new B();
C d1 = new D();
```

Données :
L'instruction B d2 = new D(); entraîne une erreur de compilation.
L'instruction a1.f(); entraîne une erreur de compilation.
L'instruction ((E)e1).f(); est valide et affiche "bye-bye" .
L'instruction ((B)b1).f(); est valide et affiche "hello" .
L'instruction ((D)b1).f(); entraîne une erreur d'exécution.

א. Dessinez un arbre d'héritage de toutes les classes, puis mentionnez dans quelles deux classes la méthode f() a été implémentée.

ב. Soit la classe Z :

```
public class Z
{
    public void g(){}
}
```

(1) Ajoutez la classe Z à l'arbre d'héritage que vous avez dessiné à l'article א de sorte que l'instruction ci-dessous soit valide :

```
Z x = new A();
```

(2) La portion de code ci-dessous est **valide**.

```
A a2 = new A();
Z z1 = new Z();
Z a3 = new A();
```

Pour chacune des instructions i-v ci-dessous, mentionnez si elle est valide ou si elle n'est pas valide.
Si l'instruction n'est pas valide, justifiez pourquoi.
Les article i-v **ne sont pas** dépendants les uns des autres.

```
i    a2 = z1;
ii   a3 = z1;
iii  ((A) z1) .g();
iv   a2.g();
v    ((A)a3).g();
```

/suite en page 21/

ג. Soit la classe Y :

```
public class Y
{
    public void m(){}
}
```

(1) Ajoutez la classe Y à l'arbre d'héritage que vous avez dessiné à l'article א de sorte que l'instruction ci-dessous soit valide (sans prendre en compte les données de l'article ב).

```
C f = new Y();
```

(2) La portion de code ci-dessous est **valide**.

```
A a2 = new A();
Y y1 = new Y();
C y2 = new Y();
```

Pour chacune des instructions i-v ci-dessous, mentionnez si elle est valide ou si elle n'est pas valide.

Si l'instruction n'est pas valide, justifiez pourquoi.

Les articles i-v **ne sont pas** dépendants les uns des autres.

i `a2 = y1;`

ii `y2 = y1;`

iii `((A)y1).m();`

iv `y2.m();`

v `((A)y2).m();`

/suite en page 22/

14. Le paiement de la taxe (Property tax) d'habitation est un paiement effectué auprès des pouvoirs locaux lié à la surface d'habitation et aux terrains annexes que les résidents possèdent. Le montant de cette taxe d'habitation est de 10 shékels par m^2 de surface destiné à l'habitation et de 0,5 shékel par m^2 de terrain supplémentaire.
Par exemple, un résident possédant une surface d'habitation de 100 m^2 et une surface de terrain de 500 m^2 paye une taxe d'habitation d'un montant de $100 \times 10 + 500 \times 0{,}5 = 1250$.
Les résidents ont été séparés en trois groupes :

- le résident d'une ville paye une taxe d'habitation pour la surface d'habitation et pour le terrain qu'il possède, et reçoit une prime de 250 shékels pour le paiement de la taxe d'habitation.
- le résident sénior d'une ville — le résident de 60 ans et plus — paye une taxe d'habitation comme le résident de la ville, et a droit en plus à une réduction en fonction de son âge : 1% de réduction pour chaque année au delà de 60 ans. La réduction est calculée après déduction de la prime du montant initial.
- le résident d'un village paye une taxe d'habitation pour la surface d'habitation et pour le terrain qu'il possède et a droit à une réduction de 10%. **Le résident du village de 60 ans et plus n'a pas droit à la réduction en fonction de l'âge.**

Si après la réduction et après la prime, le montant des taxes d'habitation est négatif, le résident sera exonéré de tout paiement (il paye 0 shékel).
On a décidé de développer un programme de calcul des taxes d'habitation et de leur prélèvement auprès des résidents. Pour cela on a défini l'interface ci-dessous :

```
interface IData
{
    public String getName(); // retourne le nom d'un résident
    public double getPropertyTax(); // retourne le montant de la taxe d'habitation.
}
```

Dans les articles א-ב ci-dessous, définissez les classes selon les principes de la programmation orientée objets conformément aux exigences suivantes :

- Définissez une fois seulement comme constantes, les données de calcul de la taxe d'habitation (paiement pour la surface de l'appartement en m^2, paiement pour la surface du terrain en m^2, niveau de réduction en pourcentage, montant de la prime en shékel)
- Définissez comme privé (private) tous les attributs qui ne sont pas constants.

א. Définissez une classe `Resident` (représentant un résident) qui implémente l'interface `IData` , et écrivez-y les méthodes de la classe (il n'est pas nécessaire d'écrire une méthode constructeur).

ב. Définissez trois classes pour chacun des groupes de résidents : `CityResident` – résident d'une ville, `SeniorCityResident` – résident sénior d'une ville, `VillageResident` – résident d'un village.
Écrivez les attributs des classes et la méthode `getPropertyTax()` qui permettra de calculer le paiement de la taxe d'habitation pour chacun des trois groupes de résidents (il n'est pas nécessaire d'écrire une méthode constructeur).

/suite en page 23/

ג. Soit une méthode qui accepte un tableau a de données des résidents, et qui affiche pour tout résident sénior de la ville, son nom et le montant de la taxe d'habitation qu'il doit payer.

```
for (int i = 0;  i < a.length; i++)
{
    if (a[i] instanceof SeniorCityResident)
    {
        System.out.println(a[i].getName()+" "+a[i].getPropertyTax());
    }
}
```

Pour chacune des déclarations des méthodes (1)-(3) ci-dessous, mentionnez si la méthode est valide ou n'est pas valide. Si la méthode n'est pas valide – justifiez pourquoi, corrigez la méthode et recopiez la méthode corrigée dans votre cahier.

(1) public static void print(Object [] a)

(2) public static void print(IData [] a)

(3) public static void print(Resident [] a)

/suite en page 24/

תכנות מונחה עצמים

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

15. Voici un projet dans lequel ont été définies les classes suivantes : A , B , C , D , E . Dans ce projet la méthode F() a été implémentée dans deux classes.
La portion de code ci-dessous est **valide**.

```
A a1 = new A();
A e1 = new E();
E c1 = new C();
C b1 = new B();
C d1 = new D();
```

Données :
L'instruction B d2 = new D(); entraîne une erreur de compilation.
L'instruction a1.F(); entraîne une erreur de compilation.
L'instruction ((E)e1).F(); est valide et affiche "bye-bye" .
L'instruction ((B)b1).F(); est valide et affiche "hello" .
L'instruction ((D)b1).F(); entraîne une erreur d'exécution.

א. Dessinez un arbre d'héritage de toutes les classes, puis mentionnez dans quelles deux classes la méthode F() a été implémentée.

ב. Soit la classe Z :

```
public class Z
{
    public void G(){}
}
```

(1) Ajoutez la classe Z à l'arbre d'héritage que vous avez dessiné à l'article א de sorte que l'instruction ci-dessous soit valide :

```
Z x = new A();
```

(2) La portion de code ci-dessous est **valide**.

```
A a2 = new A();
Z z1 = new Z();
Z a3 = new A();
```

Pour chacune des instructions i-v ci-dessous, mentionnez si elle est valide ou si elle n'est pas valide.
Si l'instruction n'est pas valide, justifiez pourquoi.
Les article i-v **ne sont pas** dépendants les uns des autres.

```
i    a2 = z1;
ii   a3 = z1;
iii  ((A)z1).G();
iv   a2.G();
v    ((A)a3).G();
```

/suite en page 25/

ג. Soit la classe Y :

```
public class Y
{
    public void M(){}
}
```

(1) Ajoutez la classe Y à l'arbre d'héritage que vous avez dessiné à l'article א de sorte que l'instruction ci-dessous soit valide (sans prendre en compte les données de l'article ב).

```
C f = new Y();
```

(2) La portion de code ci-dessous est **valide**.

```
A a2 = new A();
Y y1 = new Y();
C y2 = new Y();
```

Pour chacune des instructions i-v ci-dessous, mentionnez si elle est valide ou si elle n'est pas valide.

Si l'instruction n'est pas valide, justifiez pourquoi.

Les articles i-v **ne sont pas** dépendants les uns des autres.

i `a2 = y1;`

ii `y2 = y1;`

iii `((A)y1).M();`

iv `y2.M();`

v `((A)y2).M();`

/suite en page 26/

16. Le paiement de la taxe d'habitation (Property tax) est un paiement effectué auprès des pouvoirs locaux lié à la surface d'habitation et aux terrains annexes que les résidents possèdent. Le montant de cette taxe d'habitation est de 10 shékels par m^2 de surface destiné à l'habitation et de 0,5 shékel par m^2 de terrain supplémentaire.
Par exemple, un résident possédant une surface d'habitation de 100 m^2 et une surface de terrain de 500 m^2 paye une taxe d'habitation d'un montant de $100 \times 10 + 500 \times 0,5 = 1250$.
Les résidents ont été séparés en trois groupes :

- le résident d'une ville paye une taxe d'habitation pour la surface d'habitation et pour le terrain qu'il possède, et reçoit une prime de 250 shékels pour le paiement de la taxe d'habitation.
- le résident sénior d'une ville — le résident de 60 ans et plus — paye une taxe d'habitation comme le résident de la ville, et a droit en plus à une réduction en fonction de son âge : 1% de réduction pour chaque année au delà de 60 ans. La réduction est calculée après déduction de la prime du montant initial.
- le résident d'un village paye une taxe d'habitation pour la surface d'habitation et pour le terrain qu'il possède et a droit à une réduction de 10%. **Le résident du village de 60 ans et plus n'a pas droit à la réduction en fonction de l'âge.**

Si après la réduction et après la prime, le montant des taxes d'habitation est négatif, le résident sera exonéré de tout payement (il paye 0 shékel).
On a décidé de développer un programme de calcul des taxes d'habitation et de leur prélèvement auprès des résidents. Pour cela on a défini l'interface ci-dessous :

```
interface IData
{
    string GetName(); // retourne le nom d'un résident
    double GetPropertyTax(); // retourne le montant de la taxe d'habitation.
}
```

Dans les articles א-ב ci-dessous, définissez les classes selon les principes de la programmation orientée objets conformément aux exigences suivantes :

- Définissez une fois seulement comme constantes, les données de calcul de la taxe d'habitation (paiement pour la surface de l'appartement en m^2, paiement pour la surface du terrain en m^2, niveau de réduction en pourcentage, montant de la prime en shékel)
- Définissez comme privé (private) tous les attributs qui ne sont pas constants.

א. Définissez une classe `Resident` (représentant un résident) qui implémente l'interface `IData` , et écrivez-y les méthodes de la classe (il n'est pas nécessaire d'écrire une méthode constructeur).

ב. Définissez trois classes pour chacun des groupes de résidents : `CityResident` – résident d'une ville, `SeniorCityResident` – résident sénior d'une ville, `VillageResident` – résident d'un village.
Écrivez les attributs des classes et la méthode `GetPropertyTax()` qui permettra de calculer le paiement de la taxe d'habitation pour chacun des trois groupes de résidents (il n'est pas nécessaire d'écrire une méthode constructeur).

/suite en page 27/

ג. Soit une méthode qui accepte un tableau a de données des résidents, et qui affiche pour tout résident sénior de la ville, son nom et le montant de la taxe d'habitation qu'il doit payer.

```
for (int i = 0;  i < a.Length; i++)
{
    if (a[i] is SeniorCityResident)
    {
        Console.WriteLine (a[i].GetName()+" "+a[i].GetPropertyTax());
    }
}
```

Pour chacune des déclarations des méthodes (1)-(3) ci-dessous, mentionnez si la méthode est valide ou n'est pas valide. Si la méthode n'est pas valide – justifiez pourquoi, corrigez la méthode et recopiez la méthode corrigée dans votre cahier.

(1) public static void Print(Object [] a)

(2) public static void Print(IData [] a)

(3) public static void Print(Resident [] a)

Bonne chance!

בהצלחה!