מדינת ישראל
משרד החינוך

סוג הבחינה: בגרות
מועד הבחינה: **מועד מיוחד**, אוגוסט 2020
מספר השאלון: **899381**
תרגום לערבית (2)

מדעי המחשב

הוראות לנבחן

א. משך הבחינה: שלוש שעות.

ב. מבנה השאלון ומפתח ההערכה:
בשאלון זה שלושה פרקים.
פרק ראשון
(15×1)+(20×1) — 35 נק'
פרק שני
(20×2) — 40 נק'
פרק שלישי
(25×1) — 25 נק'
סה"כ — 100 נק'

ג. חומר עזר מותר בשימוש: כל חומר עזר, חוץ ממחשבון שיש בו אפשרות תכנות.

ד. הוראות מיוחדות:
1. את כל התוכניות 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: בתוכניות שאתה כותב לא יורדו לך נקודות, אם תכתוב אות גדולה במקום אות קטנה או להפך.

دولة إسرائيل
وزارة التّربية والتّعليم

نوع الامتحان: بچروت
موعد الامتحان: **موعد خاصّ**، آب 2020
رقم النّموذج: **899381**
ترجمة إلى العربيّة (2)

علم الحاسوب

تعليمات للممتحَن

أ. مدّة الامتحان: ثلاث ساعات.

ب. مبنى النّموذج وتوزيع الدّرجات:
في هذا النّموذج ثلاثة فصول.
الفصل الأوّل
(15×1)+(20×1) – 35 درجة
الفصل الثّاني
(20×2) – 40 درجة
الفصل الثّالث
(25×1) – 25 درجة
المجموع – 100 درجة

ج. موادّ مساعدة يُسمح استعمالها: كلّ مادّة مساعدة، عدا الحاسبة التي فيها إمكانيّة برمجة.

د. تعليمات خاصّة:
1. اكتب جميع البرامج التي يُطلب منك كتابتها بلغة حاسوب في الفصلين الأوّل والثاني، بلغة واحدة فقط – Java أو C#.
2. **سجّل على الغلاف الخارجيّ** للدفتر بأيّة **لغة** تكتب – Java أو C#.
3. **سجّل على الغلاف الخارجيّ** للدفتر، اسم **المسار الذي تعلّمتَه**. المسار هو أحد المسارات الأربعة التالية: أنظمة حاسوب وأسمبلي، مقدّمة لبحث الأداءات، موديلات حسابيّة، برمجة موجّهة كائنات.

ملاحظة: في البرامج التي تكتبها، لن تُخصم درجات إذا كتبتَ حرفًا كبيرًا بدلاً من حرف صغير أو بالعكس.

اكتب في دفتر الامتحان فقط. اكتب "مسوّدة" في بداية كلّ صفحة تستعملها مسوّدة.
كتابة أيّة مسوّدة على أوراق خارج دفتر الامتحان قد تسبّب إلغاء الامتحان.

التّعليمات في هذا النّموذج مكتوبة بصيغة المذكّر وموجّهة للممتحَنات وللممتحَنين على حدّ سواء.

ב ה צ ל ח ה ! **نتمنّى لك النّجاح!**

الأسئلة

في هذا النموذج ثلاثة فصول.
عليك الإجابة عن أسئلة من ثلاثة الفصول، حسب التعليمات في كلّ فصل.

ملاحظة: في كلّ سؤال يُطلب فيه استقبال، لا حاجة لفحص سلامة المدخَلات.
للّذين يحلّون بلغة Java – في كلّ سؤال يُطلب فيه استقبال، افترِضْ أنّ الأمر التالي مكتوب في البرنامج:

```
Scanner input = new Scanner (System.in);
```

الفصل الأوّل (35 درجة)

أجب عن السؤال 1 – إلزاميّ (15 درجة).

انتبه: لهذا السؤال صيغتان: بلغة Java (في الصفحتين 2-3) وبلغة C# (في الصفحة 4).

1. للّذين يحلّون بلغة Java

أمامك الفئة قلم رصاص **Pencil**.

Pencil	
صفة طول القلم	private int length
صفة قيمتها true إذا كان القلم مبريًّا، خلاف ذلك قيمتها false.	private boolean sharpened
عمليّة بنائيّة تتلقّى قيمًا بالنسبة لكلّ صفات القلم.	public Pencil (int length, boolean sharpened)
عمليّة تتلقّى كائنًا آخر من نمط Pencil وتُعيد true إذا كان القلم الحاليّ أطول من القلم الآخر – other، خلاف ذلك تُعيد العمليّة false. افترِضْ أنّ other ليس null.	public boolean isLonger (Pencil other)
عمليّة تُعيد true إذا كان القلم مبريًّا، خلاف ذلك تُعيد العمليّة false.	public boolean isSharpened ()

(انتبه: تكملة السؤال في الصفحة التالية.)

/يتبع في صفحة 3/

أ. طبِّق العمليّتين isLonger وَ isSharpened .

ب. أمامك قطعة برنامج مكتوبة في العمليّة main للفئة الرئيسيّة.

```
Pencil pencil1 = new Pencil (12, true);
Pencil pencil2 = new Pencil (13, false);
if (pencil1.isLonger (pencil2) && pencil1.isSharpened ())
        System.out.println ("I choose pencil1");
else {
        if (pencil2.isSharpened ())
                System.out.println ("I choose pencil2");
        else
                System.out.println ("pencil2 needs to be sharpened");
}
```

تتبّع تنفيذ قطعة البرنامج، واعرض الكائنات التي تكوّنت، واكتب المُخرَج.

/ يتبع في صفحة 4 /

للّذين يحلّون بلغة C#

أمامك الفئة قلم رصاص **Pencil**.

Pencil	
صفة طول القلم	private int length
صفة قيمتها true إذا كان القلم مبريًّا، خلاف ذلك قيمتها false .	private bool sharpened
عمليّة بنائيّة تتلقّى قيمًا بالنسبة لكلّ صفات القلم.	public Pencil (int length, bool sharpened)
عمليّة تتلقّى كائنًا آخر من نمط Pencil وتُعيد true إذا كان القلم الحاليّ أطول من القلم الآخر – other ، خلاف ذلك تُعيد العمليّة false . افترِضْ أنّ other ليس null .	public bool IsLonger (Pencil other)
عمليّة تُعيد true إذا كان القلم مبريًّا، خلاف ذلك تُعيد العمليّة false .	public bool IsSharpened ()

أ. طبِّق العمليّتين IsLonger وَ IsSharpened.

ب. أمامك قطعة برنامج مكتوبة في العمليّة Main للفئة الرئيسيّة.

```
Pencil pencil1 = new Pencil (12, true);
Pencil pencil2 = new Pencil (13, false);
if (pencil1.IsLonger (pencil2) && pencil1.IsSharpened ())
        Console.Writeln ("I choose pencil1");
else {
        if (pencil2.IsSharpened ())
                Console.Writeln ("I choose pencil2");
        else
                Console.Writeln ("pencil2 need to be sharpened");
}
```

تتبّع تنفيذ قطعة البرنامج، واعرض الكائنات التي تكوّنت، واكتب المُخرَج.

/ يتبع في صفحة 5 /

أجب عن أحد السؤالين 2-3 (20 درجة).

2. معطاة الفئة **Doctor** – طبيب، لها ثلاث صفات:

- name – اسم الطبيب من نمط نصّ.
- specialization – مجال التخصّص من نمط نصّ.
- rate – تدريج الطبيب من نمط عدد حقيقيّ، يتلقّى قيَمًا من 0 حتّى 10 .

افترِضْ أنّه تمّ لكلّ صفة في الفئة **Doctor** تعريف عمليّتَي get وَ set بلغة Java وعمليّتَي Get وَ Set بلغة C#.

أ. اكتب عمليّة بنائيّة تتلقّى قيَمًا بالنسبة لاسم الطبيب ومجال التخصّص وتبتدئ الصفة rate لصفر.

يتمّ حساب تدريج (rate) الطبيب حسب معدّل تقديرات مُعالَجيه. التقدير هو عدد صحيح بين 0 وَ 10 (بما في ذلك 0 وَ 10).

ب. اكتب عمليّة void خارجيّة بلغة Java باسم doctorRating، أو بلغة C# باسم DoctorRating، تتلقّى مصفوفة باسم doctors من نمط **Doctor** . افترِضْ أنّ قيمة التدريج – rate لحدود المصفوفة هي 0. تستقبل العمليّة، حسب ترتيب الحدود في المصفوفة، جميع التقديرات التي منحها المعالَجون لكلّ طبيب. المُدخَل بالنسبة لكلّ طبيب ينتهي عند استقبال القيمة (1−) ، وبعد القيمة (1−) يبدأ مُدخَل الطبيب التالي في المصفوفة، حتّى آخر الأطبّاء في المصفوفة (الذي ينتهي هو أيضًا عند استقبال القيمة (1−)).
تُعدِّل العمليّة تدريج كلّ طبيب وفقًا لحساب معدّل تقديرات مُعالَجيه.

مثال: بالنسبة لمصفوفة doctors – بِكبَر 3 تنتج المُدخَلات التالية:
1− ,7 ,6 ,1− ,2 ,1− ,8 ,9 ,7 ,10 (من اليسار إلى اليمين).
تُعدِّل العمليّة تدريج الطبيب الأوّل في المصفوفة (في المؤشِّر 0) لِـ 8.5، وتدريج الطبيب الثاني لِـ 2 وتدريج الطبيب الثالث لِـ 6.5 .

افترِضْ أنّه يوجد على الأقلّ تقدير واحد بالنسبة لكلّ طبيب في المصفوفة.
افترِضْ أنّ جميع الحدود في المصفوفة doctors لا تساوي null .
لا حاجة لفحص سلامة المُدخَلات.

/ يتبع في صفحة 6 /

3. حدّ في مصفوفة أحاديّة الأبعاد يُسمّى **حدّ تكافؤ** إذا كان مجموع الحدود في المصفوفة من جهته اليمنى يساوي مجموع الحدود في المصفوفة من جهته اليسرى.

للحدّ الموجود في طرف المصفوفة، لا توجد حدود عن يمينه أو يساره، وفي هذه الحالة يكون المجموع في الجهة التي لا توجد فيها حدود 0 .

مثال: في المصفوفة الأحاديّة الأبعاد بِكبَر 6 ، التي أمامك، الحدّ المُشار إليه هو **حدّ تكافؤ**، لأنّ مجموع الحدود من جهتَيْه يساوي 6.

−1	7	4	2	3	1

مثال آخر: في المصفوفة الأحاديّة الأبعاد بِكبَر 7 التي أمامك، الحدّ المُشار إليه هو **حدّ تكافؤ**، لأنّ مجموع الحدود من جهتَيْه يساوي 0 .

6	−3	4	2	0	−6	3

أ. اكتب بلغة Java أو بلغة C# عمليّة خارجيّة تتلقّى:

- مصفوفة أحاديّة الأبعاد من نمط صحيح.
- عددًا يشير إلى مكان حدّ في المصفوفة (مؤشّر)، من نمط صحيح.

افترِضْ أنّ العدد المتلقّى سليم.

تفحص العمليّة إذا كان هذا الحدّ **حدّ تكافؤ**.

إذا كان كذلك – تُعيد العمليّة true، خلاف ذلك تُعيد false .

ب. مصفوفة أحاديّة الأبعاد تُسمّى **مصفوفة تكافؤ** إذا وُجد في المصفوفة حدّ واحد على الأقلّ هو **حدّ تكافؤ**.

اكتب بلغة Java أو بلغة C# عمليّة خارجيّة تتلقّى مصفوفة أحاديّة الأبعاد من نمط صحيح.

تفحص العمليّة إذا كانت المصفوفة **مصفوفة تكافؤ**.

إذا كانت كذلك – تُعيد العمليّة true، خلاف ذلك تُعيد false .

ملاحظة: بإمكانك استعمال العمليّة التي كتبتَها في البند "أ".

/يتبع في صفحة 7/

الفصل الثاني (40 درجة)

انتبه: في كلّ سؤال يُطلب فيه تطبيق، بإمكانك استعمال عمليّات الفئات: دور وراصّة وشجرة بينارية وحلقة، بدون تطبيقها. إذا استعملتَ عمليّات إضافيّة، عليك تطبيقها.

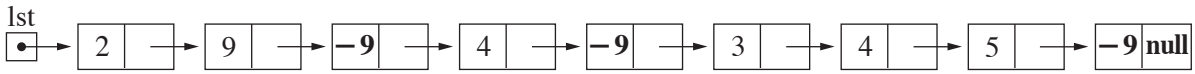
أجب عن اثنين من الأسئلة 4-6 (لكلّ سؤال – 20 درجة).

4. • "**سلسلة أعداد صحيحة موجبة**" هي سلسلة حلقات تحوي كلّ حلقة فيها عددًا صحيحًا أكبر من 0.

• "**سلسلة أرقام**" هي سلسلة حلقات تحوي كلّ حلقة فيها رقمًا بين 0 وَ 9 (بما في ذلك 0 وَ 9) أو العدد (9 –).

كلّ تسلسل أرقام في السلسلة يمثّل عددًا: الرقم الأوّل يمثّل الآحاد، والرقم الثاني يمثّل العشرات وهكذا.
بعد كلّ تسلسل أرقام يظهر العدد (9 –)، وهو يشير إلى نهاية عدد في السلسلة.

أمامك مثال لـ "سلسلة أرقام" تمثّل الأعداد: 543 ، 4 ، 92.



اكتب عمليّة خارجيّة بلغة Java باسم buildDigit أو بلغة C# باسم BuildDigit تتلقّى توجيهًا lst ليس null إلى "سلسلة أعداد صحيحة موجبة". تُعيد العمليّة "سلسلة أرقام" تمثّل الأعداد التي في "سلسلة أعداد صحيحة موجبة" حسب الترتيب.

/ يتبع في صفحة 8 /

5. نُعرِّف "بلوك" في الراصّة على أنّه تسلسل لحدّين متطابقين على الأقلّ.
اكتب بلغة Java أو بلغة C# عمليّة خارجيّة تتلقّى راصّة stk — من نمط صحيح، وتُعيد راصّة جديدة. الراصّة المُعادَة تحوي جميع الحدود التي ليست في "بلوك" في الراصّة stk.
لا توجد أهمّيّة لترتيب الحدود في الراصّة المُعادَة.

ملاحظات:
- إذا كان عدد معيّن يظهر ولكن ليس في "بلوك"، فسيتواجد في الراصّة المُعادَة حتّى إذا كان هذا العدد يظهر في "بلوك" أيضًا.
- إذا كان عدد معيّن يظهر عدّة مرّات، ليس في "بلوك"، فهو يظهر بنفس عدد المرّات في الراصّة المُعادَة.
- إذا كات الراصّة stk فارغة، تُعاد راصّة فارغة.

مثال:

الراصّة stk

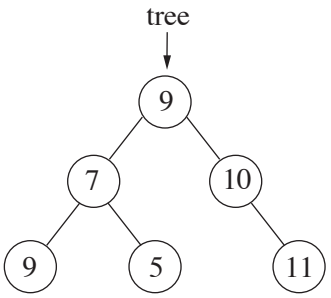
رأس الراصّة ←	14
	15
	5
	5
	0
	0
	0
	5
	3
	−4
	−4
	15

الراصّة المُعادَة

15
3
5
15
14

/ يتبع في صفحة 9 /

6. شجرة بينارية من نمط صحيح لأعداد ليست سالبة هي "**شجرة بواقٍ متساوية**" في الحالة التالية:
كمّيّة الحدود التي أعدادها تقسم على 3 مع باقي 1 تساوي كمّيّة الحدود التي أعدادها تقسم على 3 مع باقي 2 وتساوي كمّيّة الحدود التي أعدادها تقسم على 3 بدون باقٍ.
مثال لِـ "شجرة بواقٍ متساوية":



هذه الشجرة البينارية هي "شجرة بواقٍ متساوية"، لأنّ فيها عددين يقسمان على 3 بدون باقٍ (9, 9)، وعددين يقسمان على 3 مع باقي 1 (7, 10) وعددين يقسمان على 3 مع باقي 2 (5, 11).

اكتب عمليّة خارجيّة بوليانيّة بلغة Java باسم treeEqual أو بلغة C# باسم TreeEqual تتلقّى شجرة بينارية من نمط صحيح، ليست فارغة، لأعداد ليست سالبة، وتفحص إذا كانت "شجرة بواقٍ متساوية".
إذا كانت كذلك، تُعيد العمليّة true، خلاف ذلك تُعيد العمليّة false.

/ يتبع في صفحة 10 /

الفصل الثالث (25 درجة)

في هذا الفصل أسئلة في أربعة مسارات:
أنظمة حاسوب وأسمبلي، ص 10-11.
مقدّمة لبحث الأداءات، ص 12-14.
موديلات حسابيّة، ص 15.
برمجة موجّهة كائنات بلغة Java، ص 16-19؛ برمجة موجّهة كائنات بلغة #C ، ص 20-23.

أجب عن سؤال واحد في المسار الذي تعلّمتَه.

أنظمة حاسوب وأسمبلي

إذا تعلّمتَ هذا المسار، أجب عن أحد السؤالين 7-8 (25 درجة).

7. في هذا السؤال ثلاثة بنود "أ – جـ". لا توجد علاقة بين البنود. أجب عن جميعها.

أ. أمامك قطعة برنامج بلغة أسمبلي:

```
        MOV BX, 100H
        MOV SI,  0
        MOV DI, 7
        MOV AL, 4
        MOV AH, 8
AGAIN:  MOV [ BX +SI ], AL
        MOV [ BX +DI ], AH
        INC SI
        DEC DI
        CMP DI, 1
        JA AGAIN
```

تتبّع بواسطة جدول متابعة تنفيذ قطعة البرنامج. يجب أن يشمل جدول المتابعة عمودًا لكلّ واحد من العوامل التالية: AL ، AH ، BX ، SI ، DI ، BX+DI ، BX+SI . كما ويجب رسم خارطة الذاكرة في المجال الملائم.

ب. اكتب قطعة كود تفحص إذا كانت في الخازنين، AL وَ AH ، مخزونة أعداد فرديّة. إذا كانت مخزونة، تُعوِّض قطعة الكود الرقم 1 في الخازن DL ، خلاف ذلك تُعوِّض قطعة الكود الرقم 0 في الخازن DL .

جـ. ما هي قيَم الأعلام CF ، ZF ، SF ، والخازن AX بعد تنفيذ كلّ الأوامر التي أمامك؟

```
MOV AL, 20h
MOV AH, 6h
SUB AX, 0F6h
```

/ يتبع في صفحة 11 /

8. مصفوفة "**متوالية تربيعيّة**" هي مصفوفة قيمة كلّ حدّ فيها يساوي تربيع مؤشّره في المصفوفة.

مثال: المصفوفة التي أمامك هي مصفوفة "**متوالية تربيعيّة**":

المؤشّر	9	8	7	6	5	4	3	2	1	0
القيمة	81	64	49	36	25	16	9	4	1	0

في مقطع المعطيات عُرِّفت المعطيات التي أمامك:

```
ARR DB 10 DUP (?)
RANK DB ?
```

أ. اكتب برنامجًا فرعيًّا باسم POW يتلقّى عددًا صحيحًا بين 0 وَ 9 (بما في ذلك 0 وَ 9) بواسطة راصّة، ويخزن تربيع العدد في الخازن AL. لا حاجة لفحص سلامة المدخَلات.

ب. اكتب برنامجًا يفحص إذا كانت المصفوفة ARR "متوالية تربيعيّة".
إذا كانت كذلك، يعوّض البرنامج العدد 10 في المتغيّر RANK، خلاف ذلك، يعوّض البرنامج في المتغيّر RANK مؤشّر الحدّ الأوّل الذي قيمته ليست تربيع مؤشّره. عليك الاستعانة بالبرنامج الفرعيّ الذي كتبتَه في البند "أ".

مثال: المصفوفة ARR التي أمامك ليست "متوالية تربيعيّة"، لأنّ فيها حدودًا قِيَمها لا تساوي تربيع مؤشّراتها. في الحالة التي في هذا المثال، يُعوِّض البرنامج المؤشّر 4 في المتغيّر RANK، لأنّ في هذا المؤشّر يظهر الحدّ الأوّل في المصفوفة الذي لا يساوي تربيع مؤشّره.

المؤشّر	**9**	**8**	7	6	**5**	**4**	3	2	1	0
القيمة	**70**	**60**	49	36	**22**	**20**	9	4	1	0

/ يتبع في صفحة 12 /

مقدّمة لبحث الأداءات

إذا تعلّمتَ هذا المسار، أجب عن أحد السؤالين 9-10 (25 درجة) .

9. في هذا السؤال خمسة بنود "أ – هـ" . لا توجد علاقة بين البنود . أجب عن جميعها .
معطاة مسألة تخطيط خطّيّ :

$$\max\{z = x_1 + 3x_2\}$$

تخضع للاضطرارات التالية :

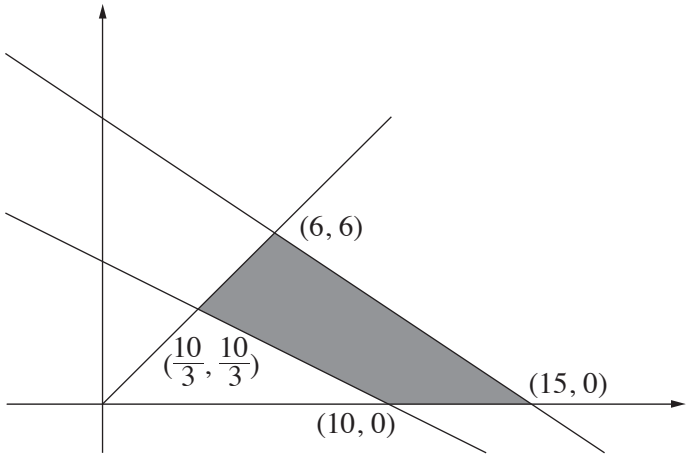
(1) $-10x_1 - 15x_2 \geq -150$

(2) $5x_1 + 10x_2 \geq 50$

(3) $x_1 - x_2 \geq 0$

(4) $x_2 \geq 0$

أمامك رسم لمجال الحلول الممكنة للمسألة المعطاة .



كلّ واحد من البنود "أ – هـ" التي في الصفحة التالية يتطرّق إلى مسألة التخطيط الخطّيّ المعطاة .
معطاة أربعة أقوال i-iv . بالنسبة لكلّ بند من البنود "أ – هـ" التي في الصفحة التالية، يوجد قول واحد فقط صحيح .

i يوجد فقط حلّ أمثل وحيد .

ii توجد لانهاية من الحلول المثلى .

iii الحلّ الأمثل ليس محصورًا .

iv لا يوجد حلّ أمثل .

/ يتبع في صفحة 13 /

بالنسبة لكلّ واحد من البنود "أ – هـ"، حدِّد أيّ قول من الأقوال i-iv صحيح. اذكر البند، وانسخ القول الصحيح إلى دفترك، وعلّل تحديدك.

- إذا اخترتَ القول i بالنسبة لأحد البنود، عليك إيجاد الحلّ الأمثل الوحيد، وقيمة دالّة الهدف في هذا الحلّ.
- إذا اخترتَ القول ii بالنسبة لأحد البنود، عليك كتابة الحلّ الأمثل العامّ للمسألة، وقيمة دالّة الهدف في مجال الحلول المثلى.

أ. أيّ قول هو الصحيح بالنسبة لمسألة التخطيط الخطّيّ المعطاة في بداية السؤال؟ علّل إجابتك.

ب. نُغيِّر فقط دالّة هدف المسألة المعطاة في بداية السؤال إلى $\min\{z = 3x_1 + 3x_2\}$.
أيّ قول هو الصحيح بالنسبة للمسألة بعد التغيير؟ علّل إجابتك.

جـ. نُغيِّر فقط دالّة هدف المسألة المعطاة في بداية السؤال إلى $\max\{z = -(3x_1 + 3x_2)\}$.
أيّ قول هو الصحيح بالنسبة للمسألة بعد التغيير؟ علّل إجابتك.

د. نُضيف الاضطرار $x_1 \geq 15$ إلى المسألة المعطاة في بداية السؤال. أيّ قول هو الصحيح بالنسبة للمسألة بعد إضافة الاضطرار؟ علّل إجابتك.

هـ. نُغيِّر فقط دالّة هدف المسألة المعطاة في بداية السؤال إلى $\max\{z = ax_1 + 3x_2\}$.
بالنسبة لأيّة قيَم لِـ a يكون $(15, 0)$ ، هو الحلّ الأمثل لمسألة التخطيط المعطاة؟ علّل إجابتك.

/ يتبع في صفحة 14 /

10. في هذا السؤال بندان، "أ – ب". لا علاقة بين البندين. أجب عن البندين.

أ. في الجدول الذي أمامك معطى حلّ أساسيّ ممكن لمسألة نقل، ومعطاة قيَم $u_3, u_2, u_1, v_3, v_2, v_1$.
هل الحلّ هو أمثل؟ علّل إجابتك.

u_i	العرض	الغايات 3	الغايات 2	الغايات 1	المصادر
0	180	17 (30)	15 (150)	14	1
-7	100	14	8 (0)	10 (100)	2
1	100	18 (100)	20	15	3
		130	150	100	الطلب
		17	15	17	v_j

ב. في الجدول الذي أمامك مسألة نقل وجزء من حلّ أساسيّ ممكن: $x_{11} = 100$.

العرض	الغايات 4	3	2	1	المصادر
300	7	8	3	5 (100)	1
500	12	10	5	9	2
200	3	7	6	8	3
	400	200	300	100	الطلب

(1) انسخ الجدول إلى دفترك، وأكمل فيه القيَم حسب طريقة الزاوية الشماليّة الغربيّة.

(2) إذا غيّرنا السعر في الخليّة (3,3) من 7 إلى 3 ، هل يتغيّر الحلّ الأساسيّ الممكن الذي وجدتَه في البند الفرعيّ (1)؟ علّل إجابتك.

(3) نُحدِّد $u_2 = 0$. أكمل في الجدول الذي كتبتَه في البند الفرعيّ (1) قيَم $u_1, u_3, v_1, v_2, v_3, v_4$.

(4) هل الحلّ الأساسيّ الممكن هو الحلّ الأمثل؟

/ يتبع في صفحة 15 /

موديلات حسابيّة

إذا تعلّمت هذا المسار أجب عن أحد السؤالين 11-12 (25 درجة) .

11. أمامك اللغة L فوق الأبجديّة {a , b} .

$$L = \{a^n b^m a^k \mid n > 0,\ m > n,\ n + k > m\}$$

ابنِ أوتومات راصّة يتلقّى اللغة L .

أمثلة لكلمات تتبع للّغة L :

abbaa, aabbbaa

أمثلة لكلمات لا تتبع للّغة L :

baa, aba, abba

12. معطاة عمليّة مكتوبة بلغة Java وبلغة C# .

العمليّة تتلقّى عددين صحيحين أكبر من 0 .

C#

```
public static int Foo (int x, int y)
{
        return (x+y)/3;
}
```

Java

```
public static int foo (int x, int y)
{
        return (x+y)/3;
}
```

اكتب آلة تيورينج تطبّق العمليّة المعطاة .

افترِضْ أنّ x + y يقسم على 3 بدون باقٍ .

مُدخَل الآلة هو العددان x , y اللذان تتلقّاهما العمليّة، وهذا المُدخَل مكتوب على الشريط من بدايته .

كلّ عدد مكتوب بشكل أونارِيّ . الإشارة # تفصل بين عدد وعدد آخر .

مثال : إذا تلقّت العمليّة 3 لِـ x وَ 3 لِـ y ، يبدو الشريط على النحو التالي :

⊢	1	1	1	#	1	1	1	△	....

المُخرَج هو القيمة التي تُعيدها العمليّة، ويُكتَب هذا المُخرَج على الشريط في مكان ما كقيمة أونارِيّة بين إشارتَي $.

مُخرَج ممكن للمثال أعلاه :

....	△	△	$	1	1	$	△	△	....

/ يتبع في صفحة 16 /

برمجة موجّهة كائنات

إذا تعلّمتَ هذا المسار وتكتب بلغة Java، أجب عن أحد السؤالين 13-14 (25 درجة) .

13. في نادي اللياقة البدنيّة "السلام" يمكن التمرّن بجميع الأجهزة الرياضيّة. بالإضافة إلى ذلك، يمكن التمرّن تمرينًا خاصًّا يمكن فيه استعمال جميع الأجهزة الرياضيّة برفقة مدرِّب شخصيّ لوقت محدَّد.

بغرض حفظ المعلومات عمّا يحدث في النادي، بُنيت الفئات التي أمامك:

مدرِّب — **Trainer**، تمرين — **Training**، تمرين خاصّ — **SpecialT** ، زبون — **Client**، إدارة — **Management** .

صفات الفئات هي وفقًا لمتطلّبات المعلومات التي أمامك:

المعلومات التي تُحفَظ عن المدرِّب:

- name — اسم المدرِّب، من نمط نصّ.
- wid — رقم العامل، من نمط صحيح.

المعلومات التي تُحفَظ عن التمرين:

- num — عدد الأجهزة الرياضيّة التي استخدمها الزبون في التمرين، من نمط صحيح.

المعلومات التي تُحفَظ عن التمرين الخاصّ:

- num — عدد الأجهزة الرياضيّة التي استخدمها الزبون في التمرين، من نمط صحيح.
- trainer — المدرِّب الذي رافق الزبون في التمرين، من نمط **Trainer** .
- time — عدد دقائق التمرين، من نمط صحيح. العدد أكبر من 0 وليس محدّدًا لِـ 60.

المعلومات التي تُحفَظ عن زبائن النادي:

- id — رقم الهويّة، من نمط نصّ.
- name — اسم الزبون، من نمط نصّ.
- visits — جميع التمرينات التي قام بها الزبون، في مصفوفة من نمط **Training** .

المعلومات التي تُحفَظ بالنسبة للإدارة:

- staff — جميع المدرِّبين الذين يعملون في النادي، في مصفوفة من نمط **Trainer** .
- clients — جميع زبائن النادي، في مصفوفة من نمط **Client** .

(انتبه: تكملة السؤال في الصفحة التالية.) / يتبع في صفحة 17 /

عليك الإجابة عن البنود التي أمامك بالطريقة الأكثر ملاءمة لمبادئ البرمجة الموجّهة كائنات.

أ. ارسم مخطّطًا هرميًّا بين الفئات.

يجب الإشارة إلى توريث بواسطة السهم: ⟶

يجب الإشارة إلى احتواء بواسطة الإشارة: ◆⟶

ب. اكتب عناوين وصفات الفئة.

افترِضْ أنّه توجد عمليّة بنائيّة وعمليّتا get وَ set في جميع الفئات.

جـ. اكتب عمليّة داخليّة في الفئة **Client** ، تُعيد مجموع كلّ الدقائق التي تمرّنها الزبون في تمرين خاصّ.
تُعيد العمليّة 0 إذا لم يتمرّن الزبون بتاتًا في تمرين خاصّ.

د. اكتب عمليّة داخليّة في الفئة **Management** ، تُعيد العدد الكلّيّ للزبائن الذين تمرّنوا في تمرين خاصّ مرّة واحدة على الأقلّ.

/ يتبع في صفحة 18 /

14. في محلّ كلّ شيء "كلّ شيء للمنزل" يبيعون منتَجات غذائيّة لا تحتاج إلى خزن في التبريد، ومنتَجات غذائيّة تحتاج إلى خزن في التبريد ومنتَجات إلكترونيّة.

بهدف إدارة مخزون المنتَجات التي في المحلّ، بُنيت الفئات التالية:

Item – مُنتَج، **FoodItem** – مُنتَج غذائيّ، **FoodRefrigerated** – مُنتَج غذائيّ يحتاج إلى خزن في التبريد، **ElectronicItem** – مُنتَج إلكترونيّ، **ItemDate**.

ItemDate هي فئة تمثّل تاريخًا وفيها 3 صفات من نمط عدد صحيح: يوم day ، شهر month ، سنة year .

صفات الفئات هي وفقًا لمتطلّبات المعلومات التي أمامك:

المعلومات التي يحفظونها بالنسبة لكلّ واحد من المنتَجات في المحلّ:

- name – اسم المنتَج، من نمط نصّ.
- catalogNumber – رقم كتالوجيّ، من نمط نصّ.
- quantity – عدد وحدات المنتَج في المخزون، من نمط صحيح.
- minQuantity – العدد الأدنى لوحدات المنتَج التي يجب أن تكون في المخزون، من نمط صحيح.

المعلومات الإضافيّة التي يحفظونها بالنسبة لجميع المنتَجات الغذائيّة:

- expiryDate – تاريخ انتهاء صلاحيّة المنتَج، من نمط **ItemDate** .

المعلومات الإضافيّة التي يحفظونها بالنسبة للمنتَجات الغذائيّة التي تحتاج إلى خزن في التبريد:

- minTemperature – درجة الحرارة الصغرى اللازمة لخزن المنتَج الغذائيّ، من نمط صحيح.
- maxTemperature – درجة الحرارة القصوى الممكنة لخزن المنتَج الغذائيّ، من نمط صحيح.

المعلومات الإضافيّة التي يحفظونها للمنتَجات الإلكترونيّة:

- guarenteeDate – تاريخ انتهاء كفالة المُصنِّع للمنتَج، من نمط **ItemDate**.

ملاحظة: المنتَجات الغذائيّة التي لا تحتاج إلى خزن في التبريد والمنتَجات الإلكترونيّة، تُحفَظ في أيّة درجة حرارة.

يمكن الافتراض أنّه توجد في جميع الفئات عمليّات بنائيّة وعمليّتا get وَ set .

(انتبه: تكملة السؤال في الصفحة التالية.) / يتبع في صفحة 19 /

أ. ارسم مخطّطًا هرميًّا بين الفئات:

Item, **ElectronicItem**, **FoodItem**, **FoodRefrigerated**, **ItemDate**.

يجب الإشارة إلى توريث بواسطة السهم: ⟶

يجب الإشارة إلى احتواء بواسطة الإشارة: ⟵◆

ب. طبِّق العمليّة الداخليّة () isMissing. تُعيد العمليّة true بالنسبة للمنتَج إذا كان عدد وحداته في المخزون أقلّ من عدد الوحدات الأدنى الذي يجب أن يكون في المخزون، خلاف ذلك تُعيد العمليّة false .

يجب أن تذكر في أيّة فئة تتواجد العمليّة.

جـ. في الفئة **Tester** عُرِّفت العمليّة:

```
public class Tester {
    public static boolean canBeStored (Item item, int temp) {
        return item.canBeStored (temp); }
}
```

تُعيد العمليّة true إذا كان بالإمكان حفظ المنتَج item في درجة حرارة temp ، خلاف ذلك تُعيد العمليّة false .

أضِف عمليّات لازمة للفئات كي تنفِّذ العمليّة المطلوب.

بالنسبة لكلّ عمليّة، اذكر في أيّة فئة تُضاف.

/ يتبع في صفحة 20 /

برمجة موجّهة كائنات

إذا تعلّمتَ هذا المسار وتكتب بلغة #C، أجب عن أحد السؤالين 15-16 (25 درجة) .

15. في نادي اللياقة البدنيّة "السلام" يمكن التمرّن بجميع الأجهزة الرياضيّة. بالإضافة إلى ذلك، يمكن التمرّن تمرينًا خاصًّا يمكن فيه استعمال جميع الأجهزة الرياضيّة برفقة مدرِّب شخصيّ لوقت محدَّد.

بغرض حفظ المعلومات عمّا يحدث في النادي، بُنيت الفئات التي أمامك:

مدرِّب — **Trainer**، تمرين — **Training**، تمرين خاصّ — **SpecialT** ، زبون — **Client**، إدارة — **Management** .

صفات الفئات هي وفقًا لمتطلّبات المعلومات التي أمامك:

المعلومات التي تُحفَظ عن المدرِّب:

- name — اسم المدرِّب، من نمط نصّ.
- wid — رقم العامل، من نمط صحيح.

المعلومات التي تُحفَظ عن التمرين:

- num — عدد الأجهزة الرياضيّة التي استخدمها الزبون في التمرين، من نمط صحيح.

المعلومات التي تُحفَظ عن التمرين الخاصّ:

- num — عدد الأجهزة الرياضيّة التي استخدمها الزبون في التمرين، من نمط صحيح.
- trainer — المدرِّب الذي رافق الزبون في التمرين، من نمط **Trainer** .
- time — عدد دقائق التمرين، من نمط صحيح. العدد أكبر من 0 وليس محدّدًا بِـ 60.

المعلومات التي تُحفَظ عن زبائن النادي:

- id — رقم الهويّة، من نمط نصّ.
- name — اسم الزبون، من نمط نصّ.
- visits — جميع التمرينات التي قام بها الزبون، في مصفوفة من نمط **Training** .

المعلومات التي تُحفَظ بالنسبة للإدارة:

- staff — جميع المدرِّبين الذين يعملون في النادي، في مصفوفة من نمط **Trainer** .
- clients — جميع زبائن النادي، في مصفوفة من نمط **Client** .

(انتبه: تكملة السؤال في الصفحة التالية.) /يتبع في صفحة 21/

عليك الإجابة عن البنود التي أمامك بالطريقة الأكثر ملاءمة لمبادئ البرمجة الموجّهة كائنات.

أ. ارسم مخطّطًا هرميًّا بين الفئات.

يجب الإشارة إلى توريث بواسطة السهم: ⟶

يجب الإشارة إلى احتواء بواسطة الإشارة: ◆⟶

ب. اكتب عناوين وصفات الفئة.

افترِضْ أنّه توجد عمليّة بنائيّة وعمليّتا Get وَ Set في جميع الفئات.

جـ. اكتب عمليّة داخليّة في الفئة **Client** ، تُعيد مجموع كلّ الدقائق التي تمرّنها الزبون في تمرين خاصّ.
تُعيد العمليّة 0 إذا لم يتمرّن الزبون بتاتًا في تمرين خاصّ.

د. اكتب عمليّة داخليّة في الفئة **Management** ، تُعيد العدد الكلّيّ للزبائن الذين تمرّنوا في تمرين خاصّ مرّة واحدة على الأقلّ.

/ يتبع في صفحة 22 /

16. في محلّ كلّ شيء "كلّ شيء للمنزل" يبيعون منتَجات غذائيّة لا تحتاج إلى خزن في التبريد، ومنتَجات غذائيّة تحتاج إلى خزن في التبريد ومنتَجات إلكترونيّة.

بهدف إدارة مخزون المنتَجات التي في المحلّ، بُنيت الفئات التالية:

Item – مُنتَج، **FoodItem** – مُنتَج غذائيّ، **FoodRefrigerated** – مُنتَج غذائيّ يحتاج إلى خزن في التبريد، **ElectronicItem** – مُنتَج إلكترونيّ، **ItemDate**.

ItemDate هي فئة تمثّل تاريخًا وفيها 3 صفات من نمط عدد صحيح: يوم day ، شهر month ، سنة year .

صفات الفئات هي وفقًا لمتطلّبات المعلومات التي أمامك:

المعلومات التي يحفظونها بالنسبة لكلّ واحد من المنتَجات في المحلّ:

- name – اسم المنتَج، من نمط نصّ.
- catalogNumber – رقم كتالوجيّ، من نمط نصّ.
- quantity – عدد وحدات المنتَج في المخزون، من نمط صحيح.
- minQuantity – العدد الأدنى لوحدات المنتَج التي يجب أن تكون في المخزون، من نمط صحيح.

المعلومات الإضافيّة التي يحفظونها بالنسبة لجميع المنتَجات الغذائيّة:

- expiryDate – تاريخ انتهاء صلاحيّة المنتَج، من نمط **ItemDate** .

المعلومات الإضافيّة التي يحفظونها بالنسبة للمنتَجات الغذائيّة التي تحتاج إلى خزن في التبريد:

- minTemperature – درجة الحرارة الصغرى اللازمة لخزن المنتَج الغذائيّ، من نمط صحيح.
- maxTemperature – درجة الحرارة القصوى الممكنة لخزن المنتَج الغذائيّ، من نمط صحيح.

المعلومات الإضافيّة التي يحفظونها للمنتَجات الإلكترونيّة:

- guarenteeDate – تاريخ انتهاء كفالة المُصنِّع للمنتَج، من نمط **ItemDate**.

ملاحظة: المنتَجات الغذائيّة التي لا تحتاج إلى خزن في التبريد والمنتَجات الإلكترونيّة، تُحفَظ في أيّة درجة حرارة.

يمكن الافتراض أنّه توجد في جميع الفئات عمليّات بنائيّة وعمليّتا Get وَ Set .

(انتبه: تكملة السؤال في الصفحة التالية.) /يتبع في صفحة 23/

أ. ارسم مخطّطًا هرميًّا بين الفئات:

Item, **ElectronicItem**, **FoodItem**, **FoodRefrigerated**, **ItemDate**.

يجب الإشارة إلى توريث بواسطة السهم: ⟶

يجب الإشارة إلى احتواء بواسطة الإشارة: ⟶◆

ب. طبِّق العمليّة الداخليّة () IsMissing. تُعيد العمليّة true بالنسبة للمنتَج إذا كان عدد وحداته في المخزون أقلّ من عدد الوحدات الأدنى الذي يجب أن يكون في المخزون، خلاف ذلك تُعيد العمليّة false.

يجب أن تذكر في أيّة فئة تتواجد العمليّة.

جـ. في الفئة **Tester** عُرِّفت العمليّة:

```
public class Tester {
    public static bool CanBeStored (Item item, int temp) {
        return item.CanBeStored (temp); }
}
```

تُعيد العمليّة true إذا كان بالإمكان حفظ المنتَج item في درجة حرارة temp، خلاف ذلك تُعيد العمليّة false.

أضِف عمليّات لازمة للفئات كي تنفِّذ العمليّة المطلوب.

بالنسبة لكلّ عمليّة، اذكر في أيّة فئة تُضاف.

ב ה צ ל ח ה!
نتمنّى لك النّجاح!

זכות היוצרים שמורה למדינת ישראל.
אין להעתיק או לפרסם אלא ברשות משרד החינוך.
حقوق الطّبع محفوظة لدولة إسرائيل.
النّسخ أو النّشر ممنوعان إلّا بإذن من وزارة التّربية والتّعليم.