מדינת ישראל
משרד החינוך

סוג הבחינה: בגרות
מועד הבחינה: **מועד חלופי**, קיץ תשפ"ה, 2025
מספר השאלון: 899271
תרגום לערבית (2)

دولة إسرائيل
وزارة التّربية والتّعليم

نوع الامتحان: بچروت
موعد الامتحان: **موعد بديل**، صيف 2025
رقم النّموذج: 899271
ترجمة إلى العربيّة (2)

מדעי המחשב

הוראות

א. משך הבחינה: שלוש שעות.

ב. מבנה השאלון ומפתח ההערכה:
בשאלון זה שני פרקים.
פרק ראשון — (25x2) — 50 נק'
פרק שני — (25x2) — 50 נק'
סך הכול — 100 נק'

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

ד. הוראות מיוחדות:
את כל התוכניות שיש לכתוב בשפת מחשב בפרקים הראשון והשני כתבו בשפה אחת בלבד — Java או C#.

הערה: לא יוּרדוּ נקודות אם תכתבו בתוכניות אות גדולה במקום אות קטנה או להפך.

علم الحاسوب

تعليمات

أ. مدّة الامتحان: ثلاث ساعات.

ب. مبنى النّموذج وتوزيع الدّرجات:
في هذا النّموذج فصلان.
الفصل الأوّل – (25x2) – 50 درجة
الفصل الثّاني – (25x2) – 50 درجة
المجموع – 100 درجة

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

د. تعليمات خاصّة:
اكتبوا جميع البرامج الّتي يجب كتابتها بلغة حاسوب في الفصلين الأوّل والثّاني، بلغة واحدة فقط – Java أو C#.

ملاحظة: لن تُخصم درجات إذا كتبتم في البرامج حرفًا كبيرًا بدلًا من حرف صغير أو بالعكس.

يجب الكتابة في دفتر الامتحان فقط. يجب كتابة "مسوّدة" في بداية كلّ صفحة تُستعمَل مسوّدة.
كتابة أيّة مسوّدة على أوراق خارج دفتر الامتحان قد تسبّب إلغاء الامتحان.

الأسئلة في هذا النّموذج ترد بصيغة الجمع، ورغم ذلك يجب على كلّ طالبة وطالب الإجابة عنها بشكل فرديّ.

ב ה צ ל ח ה !

نتمنّى لكم النّجاح!

الأسئلة

في هذا النموذج فصلان.

يجب الإجابة عن أسئلة من الفصلين، حسب التعليمات في كلّ فصل.

ملاحظة: في كلّ سؤال يُطلب فيه استقبال، لا حاجة لفحص سلامة المدخَلات.

للّذين يحلُّون بلغة Java: في كلّ سؤال يُطلب فيه استقبال، افترِضوا أنّ الأمر التالي مكتوب في البرنامج:

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

انتبهوا: في كلّ سؤال يُطلب فيه تطبيق، بإمكانكم استعمال عمليّات الفئات: دَوْر وراصّة وشجرة بيناريّة وحلقة، بدون تطبيقها. إذا استعملتم عمليّات إضافيّة، يجب تطبيقها.

الفصل الأوّل (50 درجة)

أجيبوا عن اثنين من الأسئلة 1-3. (لكلّ سؤال – 25 درجة.)

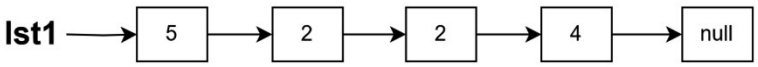
1. معطاة قائمتان لأعداد صحيحة lst1 وَ lst2.

أ. اكتبوا عمليّة خارجيّة باسم createUnion بلغة Java أو CreateUnion بلغة #C.

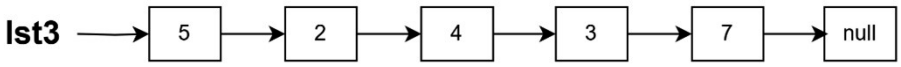
تتلقّى العمليّة القائمتين كَمُدخَل، وتُعيد قائمة جديدة تحوي جميع الأعداد التي تظهر في إحدى القائمتين على الأقلّ.

ترتيب الأعداد في القائمة الْمُعادة ليس مهمًّا، وكلّ عدد يظهر مرّة واحدة فقط.

مثال: بالنسبة لاستدعاء العمليّة مع القائمتين lst2, lst1 اللتين أمامكم:



تُعاد القائمة التالية (لا توجد أهمّيّة لترتيب المتغيّرات):



الشرح: كلّ عدد ظهر على الأقلّ في إحدى القائمتين، يظهر في القائمة الجديدة. وكذلك في القائمة الجديدة كلّ عدد يظهر مرّة واحدة فقط (بدون مضاعفات).

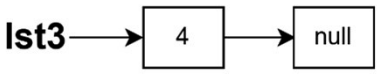
(انتبهوا: تكملة السؤال في الصفحة التالية.)

ب. اكتبوا عمليّة خارجيّة باسم intersection بلغة Java أو Intersection بلغة C#.

تتلقّى العمليّة كَمُدخَل القائمتين وتُعيد قائمة جديدة تحوي فقط الأعداد التي تظهر في lst1 و**كذلك** في lst2.

ترتيب الأعداد في القائمة الْمُعادة ليس مهمًّا، وكلّ عدد يظهر مرّة واحدة فقط. إذا لم يوجد أيّ عدد مشترك، تُعيد العمليّة قائمة فارغة.

مثال: بالنسبة لاستدعاء العمليّة مع القائمتين lst1, lst2 المعاطتين في المثال الذي في البند "أ"، تُعاد القائمة التالية:



الشرح: العدد 4 فقط، يظهر في القائمتين.

/ يتبع في صفحة 4 /

2. يُعتبَر دوران "M- مرتبطين" إذا كانت قيمة M الحدود الأخيرة في الدور الأوّل مساوية (حسب نفس الترتيب) لقيمة M الحدود الأولى في الدور الثاني.

أ. اكتبوا عمليّة باسم isMConnect بلغة Java أو IsMConnect بلغة #C، تتلقّى دورًا أوّلًا– q1 ودورًا ثانيًا– q2 (كلاهما من نمط صحيح) ومتغيّرًا من نمط صحيح M.

تُعيد العمليّة true إذا كان الدور q1 هو "M – مرتبط" بالدور q2، خلاف ذلك تُعيد العمليّة false.

ملاحظات:

- يجب الحفاظ، في نهاية العمليّة، على مبنى الدَّوْرَيْن كما تمّ تلقّيهما.
- في هذا البند، لا تستعملوا مصفوفة أو قائمة مرتبطة. الحلّ الذي يشمل مثل هذا الاستعمال لن يحصل على درجات.
- افترِضوا أنّ M هو أصغر أو يساوي طول الدَّوْرَيْن.

مثال: بالنسبة للدَّوْرَيْن اللذين أمامكم:

	رأس الدور						نهاية الدور
q1	2	5	4	4	7	**2**	**6**

	رأس الدور								نهاية الدور
q2	**2**	**6**	7	1	4	2	1	8	1

بالنسبة لِـ M = 1 يُعاد false (قيمة الحدّ الأخير في q1 -6، لا تساوي قيمة الحدّ الأوّل في q2 - 2).

بالنسبة لِـ M = 2 يُعاد true (قيمتا الحدّين الأخيرين في q1 ، تساويان، بنفس الترتيب، قيمتَي الحدّين الأوّلين في q2).

بالنسبة لِـ M = 3 يُعاد false (قِيَم ثلاثة الحدود الأخيرة في q1 -7,2,6، لا تساوي، **بنفس الترتيب**، قِيَم ثلاثة الحدود الأولى في q2 -2,6,7).

(انتبهوا: تكملة السؤال في الصفحة التالية.)

ب. اكتبوا عمليّة باسم maxConnect بلغة Java أو MaxConnect بلغة #C، تتلقّى دورًا أوّلًا q1 ودورًا ثانيًا q2 (كلاهما من نمط صحيح) .

تُعيد العمليّة الـ M الأكبر الذي بالنسبة له يتحقّق أنّ q1 هو "M – مرتبط" بِـ q2 . إذا لم يكونا مرتبطين بتاتًا، يُعاد 0 .

ملاحظة: بإمكانكم استعمال العمليّة التي في البند "أ" .

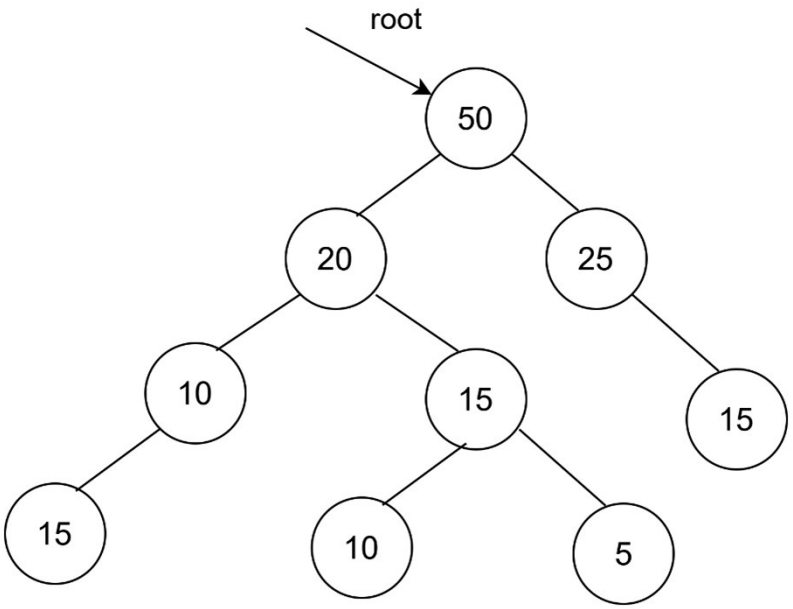
مثالان:

بالنسبة للدَّوْرَيْن من المثال الذي في البند "أ" ، والاستدعاء maxConnect(q1,q2) ، يُعاد العدد 2 (يوجد حدّان في نهاية q1 يساويان لبداية q2 . عندما نُكبِّر M لا يكونان متساويين) .

بالنسبة لنفس الدَّوْرَيْن والاستدعاء maxConnect(q2,q1) ، يُعاد العدد 0 (لا توجد حدود في نهاية q2 تلائم بداية q1 بالنسبة لأيّ M) .

/ يتبع في صفحة 6 /

3. أمامكم شجرة بينارّية جذرها root :



أ. معطاة العمليّة sod بلغة Java وَ Sod بلغة C# :

Java	public static boolean sod(BinNode<Integer> root, int num)
	{
	if (root == null)
	return false;
	if (root.getValue() == num)
	return true;
	return sod(root.getLeft(), num) || sod(root.getRight(), num);
	}

C#	public static bool Sod(BinNode<int> root, int num)
	{
	if (root == null)
	return false;
	if (root.GetValue() == num)
	return true;
	return Sod(root.GetLeft(), num) || Sod(root.GetRight(), num);
	}

(1) ما هي القيمة التي تُعاد من العمليّة بالنسبة للاستدعاء: sod(root, 25)؟ يجب تنفيذ متابعة للكود.

(2) اكتبوا استدعاءً إضافيًّا تُعاد بالنسبة له قيمة لا تساوي القيمة التي أُعيدت في البند "أ".

(3) اشرحوا ما الذي تنفذّه العمليّة.

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 7 /

ب. معطاة العمليّة secret بلغة Java وَ Secret بلغة C#:

Java	public static int secret(BinNode<Integer> root, int x)
	{
	if (!sod(root,x))
	return 0;
	if ((root.getLeft() != null && root.getLeft().getValue() == x) ||
	(root.getRight() != null && root.getRight().getValue() == x))
	{
	return root.getValue()+ secret(root.getLeft(),x)+
	secret(root.getRight(),x);
	}
	return secret(root.getLeft(),x)+secret(root.getRight(),x);
	}

C#	public static int Secret(BinNode<int> root, int x)
	{
	if (!Sod(root, x))
	return 0;
	if ((root.GetLeft() != null && root.GetLeft().GetValue() == x) ||
	(root.GetRight() != null && root.GetRight().GetValue() == x))
	{
	return root.GetValue() + Secret(root.GetLeft(), x) +
	Secret(root.GetRight(), x);
	}
	return Secret(root.GetLeft(), x) + Secret(root.GetRight(), x);
	}

(1) ما هي القيمة التي تُعاد من العمليّة بالنسبة للاستدعاء: Secret(root, 10)؟ يجب تنفيذ متابعة. لا حاجة لتنفيذ متابعة لاستدعاء العمليّة sod/Sod، وإنّما فقط ذكر القيمة التي تُعاد من العمليّة.

(2) ما هي القيمة التي تُعاد من العمليّة بالنسبة للاستدعاء: Secret(root, 5) ؟ لا حاجة لتنفيذ متابعة.

(3) اكتبوا استدعاءً إضافيًّا تُعاد بالنسبة له القيمة 50. لا حاجة لتنفيذ متابعة.

(4) اشرحوا ما الذي تنفّذه العمليّة.

/ يتبع في صفحة 8 /

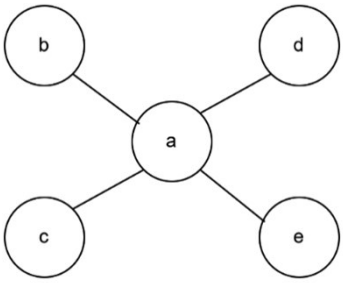
الفصل الثاني (50 درجة)

في هذا الفصل أسئلة في ثلاثة مسارات:
ألغوريثمات، الأسئلة 4-6.
موديلات حسابيّة، الأسئلة 7-9.
برمجة موجّهة كائنات، الأسئلة 10-12 بلغة Java ، تليها الأسئلة 10-12 بلغة #C.
يجب الإجابة عن سؤالين في المسار الذي تعلّمتموه فقط. (لكلّ سؤال – 25 درجة.)

ألغوريثمات

4. في هذ السؤال بندان، "أ – ب"، لا علاقة بينهما. أجيبوا عن البندين.

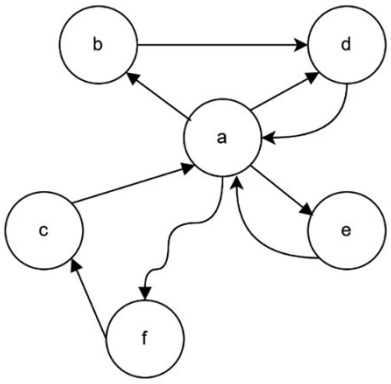
أ. أمامكم الرسم البيانيّ G=(V,E) وهو ليس موجَّهًا:



i. هل الرسم البيانيّ قابل للرَّبط؟ علِّلوا.
ii. هل الرسم البيانيّ ذو جانبين؟ علِّلوا.
iii. هل الرسم البيانيّ كامل (مكتمل)؟ علِّلوا.
iv. هل الرسم البيانيّ هو شجرة؟ علِّلوا.

(انتبهوا: تكملة السؤال في الصفحة التالية.)

ب. عرَّفَ مهندس البلديّة الانتقال بين الأحياء a,b,c,d,e,f ، حسب الرسم البيانيّ التالي:



i. ارسموا مصفوفة مجاوَرات.

ii. هل الرسم البيانيّ هو رسم بيانيّ قابل للرَّبط قويّ؟ علِّلوا.

iii. يرغب أحد سكّان المدينة في الوصول من حيّ معيّن إلى حيّ آخر. لهذا الغرض، عليه أن يفحص هل هناك مسار بين الحيَّيْن، وإذا وُجِد مسار كهذا، فهو يريد أن يجد مسارًا ما يصل إلى ذلك الحيّ. ما هو الألغوريثم الذي عليه استعماله لهذه الأغراض؟ اشرحوا.

iv. شغِّلوا الألغوريثم الذي كتبتموه في البند السابق لإيجاد المسار من الحيّ b إلى الحيّ c. اكتبوا المسار الذي وجدتموه.

يجب تنفيذ متابعة مفصَّلة في كلّ مرحلة حسب الألغوريثم الذي شغّلتموه.

5. أمامكم سبعة ادّعاءات. اختاروا خمسة منها. بالنسبة لكلّ واحد من الادّعاءات التي اخترتموها، اكتبوا رقم الادّعاء في دفتركم، واذكروا هل هو صحيح أم غير صحيح.
إذا ذكرتم بأنّ الادّعاء صحيح – علِّلوا لماذا، وإذا ذكرتم بأنّ الادّعاء غير صحيح – أعطوا مثالًا مناقضًا أو علِّلوا لماذا.

(1) في الرسم البيانيّ غير الموجَّه، إذا وُجِدت أقواس تربط بين كلّ رأسين، عندها يكون الرسم البيانيّ دائمًا قابلًا للرَّبط.

(2) الرسم البيانيّ الموجَّه الذي فيه n رؤوس وأكثر من n-1 أقواس، يحوي دائمًا دائرة.

(3) في الرسم البيانيّ الموجَّه G الذي فيه n رؤوس وَ m أقواس، إذا كان $m < n-1$ عندها يكون الرسم البيانيّ دائمًا قابلًا للرَّبط.

(4) ألغوريثم ديكسترا هو الاختيار المفضَّل عندما يحوي الرسم البيانيّ أقواسًا ذات وزن سالب، لأنّه أسرع من ألغوريثم بلمان – فورد.

(5) الغابة هي رسم بيانيّ لا توجد فيه دوائر، وهو بالضرورة غير قابل للرَّبط.

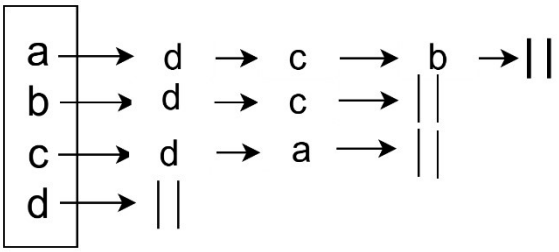
(6) يمكن استعمال ألغوريثم بلمان – فورد لإيجاد مسار قصير في رسم بيانيّ موجَّه يحوي أقواسًا مع وزن سالب، لكن ليس مع دائرة سالبة.

(7) الرسم البيانيّ G الموجَّه الذي فيه درجات جميع الرؤوس هي أكبر من 0 هو دائمًا قابل للرَّبط.

/ يتبع في صفحة 10 /

6. في هذ السؤال بندان، "أ – ب"، لا علاقة بينهما. أجيبوا عن البندين.

أ. معطى الرسم البيانيّ G = (V,E) الموجَّه، الممثَّل بواسطة قائمة المجاوَرات التالية:



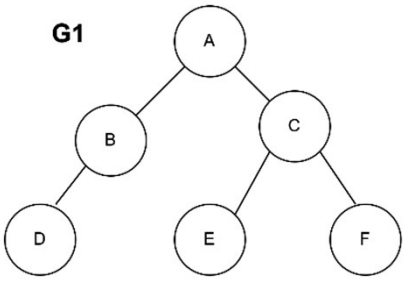
(1) ارسموا الرسم البيانيّ G الممثَّل بواسطة قائمة المجاوَرات.

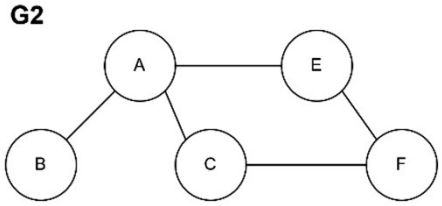
(2) شغِّلوا ألغوريثم مسح عميق(DFS) على الرسم البيانيّ G ابتداءً من العقدة a. ارسموا الشجرة الامتداديّة DFS.

(3) شغِّلوا ألغوريثم مسح عرضيّ (BFS) على الرسم البيانيّ G ابتداءً من العقدة c. ارسموا الشجرة الامتداديّة BFS.

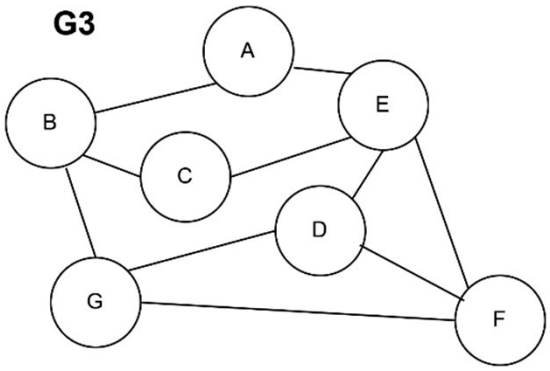
ب. الرسم البيانيّ ذو اتّجاهين هو رسم بيانيّ يمكن فيه تقسيم العُقَد التي فيه إلى مجموعتين غريبتين، بحيث لا يوجد قوس بين عقدتين تابعتين لنفس المجموعة.

(1) أمامكم رسمان بيانيّان ذوا جانبين. بيِّنوا مجموعتَي العُقَد بالنسبة لكلّ رسم بيانيّ.





(2) أمامكم رسم بيانيّ ليس ذا اتّجاهين. ما هو عدد الأقواس الأدنى التي يجب إزالتها كي يكون الرسم البيانيّ ذا جانبين؟ اكتبوا أيّة أقواس يجب إزالتها، واعرضوا المجموعتين.



/ يتبع في صفحة 11 /

موديلات حسابيّة

7. أمامكم اللغتان L_1 وَ L_2 فوق الأبجديّة $\{0,1\}$:

$L_1 = \{0^k 1^n 0^m \mid k,n,m > 0, n + m \geq k\}$

$L_2 = \{0^k 1^n \mid n,k > 0, n \bmod 2 = k \bmod 2\}$

أ. اكتبوا كلمة بطول 6 تتبع للّغة L_1 وكلمة بطول 6 تتبع للّغة L_2.

ب. إذا كانت اللغة L_1 نظاميّة، يجب بناء أوتومات نهائيّ محدود ليس كاملًا يتلقّى اللغة. إذا كانت اللغة غير نظاميّة، يجب بناء أوتومات راصّة محدود يتلقّى اللغة.

جـ. إذا كانت اللغة L_2 نظاميّة، يجب بناء أوتومات نهائيّ محدود ليس كاملًا يتلقّى اللغة. إذا كانت اللغة غير نظاميّة، يجب بناء أوتومات راصّة محدود يتلقّى اللغة.

8. في هذ السؤال بندان، "أ – ب"، لا علاقة بينهما. أجيبوا عن البندين.

أ. معطاة ثلاث لغات L_1 ، L_2 ، L_3 فوق الأبجديّة $\{a, b\}$.

L_1 = {كلّ الكلمات التي فيها عدد مرّات ظهور الحرف a والحرف b متساوٍ}

L_2 = {كلّ الكلمات التي فيها عدد مرّات ظهور الحرف a هو مرّتان على الأقلّ}

$L_3 = L_1 \cap L_2$

(1) اكتبوا كلمة بطول 6 أو أكثر تتبع للّغة L_1.

(2) اكتبوا كلمة بطول 6 أو أكثر تتبع للّغة L_2.

(3) عرِّفوا بالكلمات اللغة L_3.

(4) ما هو طول الكلمة الصغرى في كلّ واحدة من اللغات؟

ب. معطاة اللُّغة L فوق الأبجديّة $\{0, 1\}$، الكلمات التي فيها تحقِّق الشرطين التاليين:

- طول جميع الكلمات في اللغة هو 2 على الأقلّ.
- التسلسل 000 يظهر على الأكثر مرّة واحدة فقط في الكلمة.

أمثلة لكلمات في اللغة:

00, 01010, 101000100

أمثلة لكلمات ليست في اللغة:

0000, 1, 1000110001

برهِنوا أنّ اللغة L نظاميّة. يوصى باستعمال صفات الانغلاق.

/ يتبع في صفحة 12 /

9. معطاة العمليّة compare بلغة java وَ Compare بلغة C# :

Java	**C#**

public static int compare(int n1, int n2)
{
if (n1>=n2)
return 1;
return 2;
}
	public static int Compare(int n1, int n2)
	{
	if (n1 >= n2)
	return 1;
	return 2;
	}

بنوا آلة تيورينچ تُطبِّق العمليّة.
تتلقّى الآلة عددين صحيحين أكبر من 0 مكتوبين في بداية الشريط وممثَّلين بلغة أونارّية (بواسطة الرقم 1 فقط). بين العددين تظهر الإشارة #. يجب كتابة القيمة الْمُعادة من العمليّة بين إشارتَي $ في مكان ما على الشريط.

مثال، قبل تشغيل الآلة:

⊢	1	1	#	1	1	1	△	△	△

بعد تشغيل الآلة:

⊢	.	.	$	1	1	$	.

الشرح: العدد الأوّل $n_1 = 2$ والعدد الثاني $n_2 = 3$، لذلك العدد الذي تُعيده الآلة هو 2 (11 بلغة أونارّية).

/ يتبع في صفحة 13 /

برمجة موجّهة كائنات بلغة Java

10. أمامكم 3 فئات Y، Z ، Tester :

```
public class Y{
   public Y(){
       System.out.println("ctor Y");
   }
   public void do1(int x){
       System.out.println("Y do1 int "+x);
   }
   public void do1(String str){
       System.out.println("Y do1 String "+str);
   }
   public void do2(int x){
       System.out.println("Y do2 "+x);
       do1(x);
   }
}

public class Z extends Y{
   public Z(){
       System.out.println("ctor Z");
   }
   public void do1(int x){
       System.out.println("Z do1 int "+x);
   }
   public void do2(){
       System.out.println("Z do2");
       do1(2);
   }
   public void do3(int x){
       System.out.println("Z do3 "+x);
       do1(3);
   }
}
public class Tester{
   public static void main(String[] args)
   {

   }
}
```

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 14 /

أمامكم 12 قطعة كود. اختاروا 10 منها.
عوِّضوا كلّ واحدة من قِطَع الكود التي اخترتموها في العمليّة main في الفئة Tester. يجب أن تكتبوا في دفتر الامتحان رقم القطعة التي اخترتموها وأن تذكروا هل الكود سليم أم غير سليم.
إذا كان الكود سليمًا – اكتبوا الْمُخرَج، وإذا كان غير سليم – اشرحوا الخطأ.
ملاحظة: لا توجد علاقة بين قِطَع الكود. أي أنّه يجب التطرّق إلى كلّ قطعة كود وكأنّها الوحيدة في العمليّة.

1. Y y = new Y(); y.do1(2);	2. Z z = new Z(); z.do1("str");	3. Z z = new Z(); Y y = z; y.do1("str");
4. Y y = new Y(); Z z = y; z.do1(2);	5. Y y = new Y(); Z z = (Z)y; z.do1(2);	6. Y yz = new Z(); yz.do1("str");
7. Y yz = new Z(); yz.do3(4);	8. Z z = new Z(); z.do2();	9. Z yz = new Y(); yz.do2();
10. Y yz = new Z(); yz.do2();	11. Y yz = new Z(); ((Z)yz).do2();	12. Y yz = new Z(); ((Z)yz).do1("hh");

/ يتبع في صفحة 15 /

11. في محلّ لبيع الكراسي طوَّروا منظومة محوسَبة فيها الفئات التالية:
Date، **Chair** ، **ComputerChair** ، **GardenChair** ، **BarStool**، كما هو مفصَّل فيما يلي:
فيما يلي تفصيل صفات الفئات:

- **Date** – التاريخ

صفات الفئة:

- month – الشهر، من نمط صحيح.
- year – السنة، من نمط صحيح.

- **Chair** – كرسيّ

صفات الفئة:

- price – السعر، من نمط صحيح.
- createDate – تاريخ الإنتاج، من نمط Date

- **ComputerChair** – كرسيّ حاسوب

صفات الفئة:

- price – السعر، من نمط صحيح.
- wheels – عدد العجلات، من نمط صحيح.
- adjustHeight – مستوى رَفْع الكرسيّ، من نمط صحيح.
- createDate – تاريخ الإنتاج، من نمط Date

- **GardenChair** – كرسيّ حديقة

صفات الفئة:

- price – السعر، من نمط صحيح.
- isFoldable – هل يمكن ثَنْيُه، من نمط بولیانيّ.
- createDate – تاريخ الإنتاج، من نمط Date

- **BarStool** – كرسيّ مقهًى

صفات الفئة:

- price – السعر، من نمط صحيح.
- adjustHeight – مستوى رَفْع المقعد، من نمط صحيح.
- createDate – تاريخ الإنتاج، من نمط Date

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 16 /

أ. **(1)** ارسموا مخطَّطًا هرميًّا يصف العلاقة بين فئات المنظومة المحوسَبة.

يجب الإشارة إلى توريث بواسطة السهم

يجب الإشارة إلى احتواء بواسطة الإشارة

(2) اكتبوا عنوان الفئات وصفاتها. افترِضوا أنّ العمليّتين get وَ set موجودتان في جميع صفات الفئات ولا حاجة لتطبيقهما.

معطاة العمليّة البنائيّة للفئة Chair:

```
public Chair(int price, Date createDate)
```

لا حاجة لتطبيق العمليّة.

(3) أمامكم عنوان العمليّة البنائيّة للفئة ComputerChair. تتلقّى العمليّة السعر وعدد العجلات ومستوى رَفْع المقعد وتاريخ الإنتاج.

```
public ComputerChair(int price, int wheels, int height, Date date)
```

طبِّقوا العمليّة البنائيّة.

ب. معطاة الفئة Order – طلبيّة.

صفات الفئة:

- chairs – مصفوفة كراسٍ، من نمط Chair. كمّيّة الكراسي القصوى في الطلبيّة – 100.
- current – كمّيّة الكراسي في الطلبيّة الحاليّة.

يمنح المحلّ تخفيضًا على الطلبيّة حسب كمّيّة الكراسي في الطلبيّة، بالطريقة التالية:

- إذا كان عدد كراسي الحاسوب في الطلبيّة أكبر من 10، يُمنَح تخفيض بنسبة 20% على جميع كراسي الحاسوب التي في الطلبيّة.
- إذا كان عدد كراسي الحديقة في الطلبيّة أكبر من 15، يُمنَح تخفيض بنسبة 25% على جميع كراسي الحديقة التي في الطلبيّة.

اكتبوا عمليّة باسم totalPrice في الفئة Order تُعيد ثمن الطلبيّة. بالإضافة إلى ذلك، تطبع العمليّة تفاصيل الكراسي في الطلبيّة (حسب صفاتها).

/ يتبع في صفحة 17 /

12. أمامكم الفئتان One وَ Two :

```
public class One
{
   private static int num = 0;
   protected int x;

   public One (int n)
   {
        this.x = n;
        num += n;
        System.out.println ("One 1");
   }

   public One (int n1, int n2)
   {
        this.x = n1+n2;
        num += n1;
        System.out.println ("One 2");
   }

   public int getNum()
   {
        return num;
   }

   public int getX()
   {
        return x;
   }

   public int mul(int n)
   {
        return this.x * n;
   }

   public void add(One other)
   {
        this.x += other.x;
        System.out.println("x = "+ this.x);
   }
}
```

```
public class Two extends One
{
   private int y;
   public Two (){
      super(10);
      this.y = 12;
   }
   public Two (int num)
   {
      super (num/2);
      this.y = num;
      System.out.println ("Second");
   }

   public int mul()
   {
      return this.x * this.y;
   }

   public void add (One other)
   {
        if (other instanceof Two)
             this.y += ((Two)other).y;
        System.out.println("y = "+ this.y);
   }
}
```

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 18 /

معطاة الفئة Tester:

```
public class Tester
{
public static void main(String[] args)
{
One one1 = new One (1);
One one2 = new One (2, 3);
One oneTwo = new Two (4);
Two two1 = new Two ();
Two two2 = new Two (6);
// ***
}
}
```

أ. ارسموا الكائنات التي تكوّنت في العمليّة main، واكتبوا مُخرَج العمليّة.

ب. عوِّضوا كلّ واحد من الأوامر 1–10 التالية في العمليّة main في المكان الْمُشار إليه أعلاه بِـ***
اكتبوا في الدفتر رقم الأمر، واذكروا هل الكود سليم أم غير سليم.
إذا كان الكود سليمًا– اكتبوا الْمُخرَج، وإذا كان غير سليم – اشرحوا لماذا.
ملاحظة: لا توجد علاقة بين الأوامر. أي أنّه يجب التطرّق إلى كلّ أمر وكأنّه الوحيد في العمليّة.

1. System.out.println ("Total = " + one1.getNum());
2. System.out.println ("Total = " + two2.getNum());
3. System.out.println ("mul = " + one1.mul(1));
4. System.out.println ("mul = " + oneTwo.mul());
5. System.out.println ("mul = " + two1.mul());
6. one1.add(oneTwo);
7. two1.add(oneTwo);
8. ((Two)one1).mul();
9. two1 = new One(1,2);
10. ((One)two1).mul(1);

/ يتبع في صفحة 19 /

برمجة موجّهة كائنات بلغة #C

10. أمامكم 3 فئات Y، Z ، Tester :

```
public class Y
{
   public Y()
   {
      Console.WriteLine("ctor Y");
   }
   public virtual void Do1(int x)
   {
      Console.WriteLine("Y Do1 int " + x);
   }
   public void Do1(string str)
   {
      Console.WriteLine("Y Do1 String " + str);
   }
   public void Do2(int x)
   {
      Console.WriteLine("Y Do2 " + x);
      Do1(x);
   }
}

public class Z: Y
{
   public Z()
   {
      Console.WriteLine("ctor Z");
   }
   public override void Do1(int x)
   {
      Console.WriteLine("Z Do1 int " + x);
   }
   public void Do2()
   {
      Console.WriteLine("Z Do2");
      Do1(2);
   }
   public void Do3(int x)
   {
      Console.WriteLine("Z Do3 " + x);
      Do1(3);
   }
}
public class Tester{
   public static void Main(string[] args)
   {

   }
}
```

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 20 /

أمامكم 12 قطعة كود. اختاروا 10 منها.
عوِّضوا كلّ واحدة من قِطَع الكود التي اخترتموها في العمليّة Main في الفئة Tester. يجب أن تكتبوا في دفتر الامتحان رقم القطعة التي اخترتموها وأن تذكروا هل الكود سليم أم غير سليم.
إذا كان الكود سليمًا – اكتبوا الْمُخرَج، وإذا كان غير سليم – اشرحوا الخطأ.
ملاحظة: لا توجد علاقة بين قِطَع الكود. أي أنّه يجب التطرّق إلى كلّ قطعة كود وكأنّها الوحيدة في العمليّة.

1. Y y = new Y(); y.Do1(2);	2. Z z = new Z(); z.Do1("str");	3. Z z = new Z(); Y y = z; y.Do1("str");
4. Y y = new Y(); Z z = y; z.Do1(2);	5. Y y = new Y(); Z z = (Z)y; z.Do1(2);	6. Y yz = new Z(); yz.Do1("str");
7. Y yz = new Z(); yz.Do3(4);	8. Z z = new Z(); z.Do2();	9. Z yz = new Y(); yz.Do2();
10. Y yz = new Z(); yz.Do2();	11. Y yz = new Z(); ((Z)yz).Do2();	12. Y yz = new Z(); ((Z)yz).Do1("hh");

/ يتبع في صفحة 21 /

11. في محلّ لبيع الكراسي طوَّروا منظومة محوسَبة فيها الفئات التالية:
Date، **Chair** ، **ComputerChair** ، **GardenChair** ، **BarStool**، كما هو مفصَّل فيما يلي:

فيما يلي تفصيل صفات الفئات:

- **Date** – التاريخ

صفات الفئة:

- month – الشهر، من نمط صحيح.
- year – السنة، من نمط صحيح.

- **Chair** – كرسيّ

صفات الفئة:

- price – السعر، من نمط صحيح.
- createDate – تاريخ الإنتاج، من نمط Date

- **ComputerChair** – كرسيّ حاسوب

صفات الفئة:

- price – السعر، من نمط صحيح.
- wheels – عدد العجلات، من نمط صحيح.
- adjustHeight – مستوى رَفْع الكرسيّ، من نمط صحيح.
- createDate – تاريخ الإنتاج، من نمط Date

- **GardenChair** – كرسيّ حديقة

صفات الفئة:

- price – السعر، من نمط صحيح.
- isFoldable – هل يمكن ثَنْيُه، من نمط بولياني.
- createDate – تاريخ الإنتاج، من نمط Date

- **BarStool** – كرسيّ مقهًى

صفات الفئة:

- price – السعر، من نمط صحيح.
- adjustHeight – مستوى رَفْع المقعد، من نمط صحيح.
- createDate – تاريخ الإنتاج، من نمط Date

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 22 /

أ. **(1)** ارسموا مخطَّطًا هرميًّا يصف العلاقة بين فئات المنظومة المحوسَبة.

يجب الإشارة إلى توريث بواسطة السهم

يجب الإشارة إلى احتواء بواسطة الإشارة

(2) اكتبوا عنوان الفئات وصفاتها. افترِضوا أنّ العمليّتين get وَ set موجودتان في جميع صفات الفئات ولا حاجة لتطبيقهما.

معطاة العمليّة البنائيّة للفئة Chair:

```
public Chair(int price, Date createDate)
```

لا حاجة لتطبيق العمليّة.

(3) أمامكم عنوان العمليّة البنائيّة للفئة ComputerChair. تتلقّى العمليّة السعر وعدد العجلات ومستوى رَفْع المقعد وتاريخ الإنتاج.

```
public ComputerChair(int price, int wheels, int height, Date date)
```

طبِّقوا العمليّة البنائيّة.

ب. معطاة الفئة Order – طلبيّة.

صفات الفئة:

- chairs – مصفوفة كراسٍ، من نمط Chair. كمّيّة الكراسي القصوى في الطلبيّة – 100.
- current – كمّيّة الكراسي في الطلبيّة الحاليّة.

يمنح المحلّ تخفيضًا على الطلبيّة حسب كمّيّة الكراسي في الطلبيّة، بالطريقة التالية:

- إذا كان عدد كراسي الحاسوب في الطلبيّة أكبر من 10، يُمنَح تخفيض بنسبة 20% على جميع كراسي الحاسوب التي في الطلبيّة.
- إذا كان عدد كراسي الحديقة في الطلبيّة أكبر من 15، يُمنَح تخفيض بنسبة 25% على جميع كراسي الحديقة التي في الطلبيّة.

اكتبوا عمليّة باسم TotalPrice في الفئة Order تُعيد ثمن الطلبيّة. بالإضافة إلى ذلك، تطبع العمليّة تفاصيل الكراسي في الطلبيّة (حسب صفاتها).

/ يتبع في صفحة 23 /

12. أمامكم الفئتان One وَ Two :

```
public class One
{
   private static int num = 0;
   protected int x;

   public One(int n)
   {
       this.x = n;
       num += n;
       Console.WriteLine("One 1");
   }

   public One(int n1, int n2)
   {
       this.x = n1 + n2;
       num += n1;
       Console.WriteLine("One 2");
   }

   public int GetNum()
   {
       return num;
   }

   public int GetX()
   {
       return x;
   }

   public int Mul(int n)
   {
       return this.x * n;
   }

   public virtual void Add(One other)
   {
       this.x += other.x;
       Console.WriteLine("x = " + this.x);
   }
}
```

```
public class Two : One
{
   private int y;
   public Two():base(10)
   {
       this.y = 12;
   }
   public Two(int num):base(num/2)
   {
       this.y = num;
       Console.WriteLine("Second");
   }

   public int Mul()
   {
       return this.x * this.y;
   }

   public override void Add(One other)
   {
       if (other is Two)
           this.y += ((Two)other).y;
       Console.WriteLine("y = " +
this.y);
   }
}
```

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 24 /

معطاة الفئة Tester:

```
public class Tester
{
   public static void Main(string[] args)
   {
      One one1 = new One (1);
      One one2 = new One (2, 3);
      One oneTwo = new Two (4);
      Two two1 = new Two ();
      Two two2 = new Two (6);
      // ***
   }
}
```

أ. ارسموا الكائنات التي تكوّنت في العمليّة Main، واكتبوا مُخرَج العمليّة.

ب. عوِّضوا كلّ واحد من الأوامر 1–10 التالية في العمليّة Main في المكان الْمُشار إليه أعلاه بِـ***
اكتبوا في الدفتر رقم الأمر، واذكروا هل الكود سليم أم غير سليم.
إذا كان الكود سليمًا– اكتبوا الْمُخرَج، وإذا كان غير سليم – اشرحوا لماذا.
ملاحظة: لا توجد علاقة بين الأوامر. أي أنّه يجب التطرّق إلى كلّ أمر وكأنّه الوحيد في العمليّة.

1. Console.WriteLine ("Total = " + one1.GetNum());
2. Console.WriteLine ("Total = " + two2.GetNum());
3. Console.WriteLine ("Mul = " + one1.Mul(1));
4. Console.WriteLine ("Mul = " + oneTwo.Mul());
5. Console.WriteLine ("Mul = " + two1.Mul());
6. one1.Add(oneTwo);
7. two1.Add(oneTwo);
8. ((Two)one1).Mul();
9. two1 = new One(1,2);
10. ((One)two1).Mul(1);

ב ה צ ל ח ה!
نتمنّى لكم 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.