מדינת ישראל

משרד החינוך

סוג הבחינה: א. בגרות לבתי"ס על־יסודיים
ב. בגרות לנבחנים חיצוניים

מועד הבחינה: קיץ תשע"ו, 2016

מספר השאלון: **899222, 602**

תרגום לערבית (2)

מדעי המחשב

2 יחידות לימוד

הוראות לנבחן

א. משך הבחינה: שלוש שעות.

ב. מבנה השאלון ומפתח ההערכה:
בשאלון זה שלושה פרקים.
פרק ראשון: יש לענות על חמש השאלות 1-5,
לכל שאלה — 10 נק' (10×5) — 50 נק'
פרק שני: יש לענות על שתיים מהשאלות 6-8,
לכל שאלה — 15 נק' (15×2) — 30 נק'
פרק שלישי: יש לענות על אחת מהשאלות 9-10,
לשאלה — 20 נק' (20×1) — 20 נק'
סה"כ — 100 נק'

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

ד. הוראות מיוחדות:
(1) כתוב בשפה אחת בלבד את כל התכניות שאתה נדרש לכתוב.
(2) **רשום על הכריכה החיצונית** של המחברת את השפה שבה אתה כותב — Java **או** C# .
הערה: בתכניות שאתה כותב לא יורדו לך נקודות, אם תכתוב אות גדולה במקום אות קטנה או להפך.

دولة إسرائيل

وزارة التّربية والتّعليم

نوع الامتحان: أ. بجروت للمدارس الثّانويّة
ب. بجروت للممتحَنين الخارجيّين

موعد الامتحان: صيف 2016

رقم النّموذج: **899222، 602**

ترجمة إلى العربيّة (2)

علم الحاسوب

وحدتان تعليميّتان

تعليمات للممتحَن

أ. مدّة الامتحان: ثلاث ساعات.

ب. مبنى النّموذج وتوزيع الدّرجات:
في هذا النموذج ثلاثة فصول.
الفصل الأوّل: يجب الإجابة عن خمسة الأسئلة 1-5،
لكلّ سؤال – 10 درجات (10×5) – 50 درجة
الفصل الثاني: يجب الإجابة عن اثنين من الأسئلة 6-8،
لكلّ سؤال – 15 درجة (15×2) – 30 درجة
الفصل الثالث: يجب الإجابة عن أحد السؤالين 9-10،
للسؤال – 20 درجة (20×1) – 20 درجة
المجموع – 100 درجة

جـ. موادّ مساعدة يُسمح استعمالها: أيّة مادّة مساعِدة، عدا الحاسوب الذي يمكن برمجته.

د. تعليمات خاصّة:
(1) اكتب جميع البرامج التي يُطلب منك كتابتها، بلغة واحدة فقط.
(2) **اكتب على الغلاف الخارجيّ** لدفتر الامتحان اللغة التي تكتب بها – Java أو C# .
ملاحظة: في البرامج التي تكتبها لن تُخصم درجات إذا كتبت حرفًا كبيرًا مكان حرف صغير أو بالعكس.

اكتب في دفتر الامتحان فقط، في صفحات خاصّة، كلّ ما تريد كتابته مسوّدة (رؤوس أقلام، عمليّات حسابيّة، وما شابه).
اكتب كلمة "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.

ب ה צ ל ח ה ! **نتمنّى لك النّجاح!**

الأسئلة

انتبه: عليك كتابة جميع البرامج التي يُطلب منك كتابتها، بلغة واحدة فقط.
اكتب على الغلاف الخارجيّ لدفتر الامتحان اللغة التي تكتب بها – Java أو C#.
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 (50 درجة)

أجب عن خمسة الأسئلة 1-5 (لكلّ سؤال – 10 درجات).

1. أمامك الفئة **Student** التي تمثّل طالبًا. للفئة صفتان: الاسم – name، وسنة الميلاد – year.
الفئة مكتوبة بلغة Java وبلغة C#.
تطبيق العمليّة البنائيّة ناقص في الفئة.

Java

```
public class Student
{
    private String name;
    private int year;
    public Student(String name, int year) {…}
}
```

C#

```
public class Student
{
    private string name;
    private int year;
    public Student(string name, int year) {…}
}
```

أ. أكمل بلغة Java أو بلغة C# تطبيق العمليّة البنائيّة للفئة **Student**.

ب. افترض أنّه معطاة عمليّة رئيسيّة في الفئة Program. اكتب بلغة Java أو بلغة C# في العمليّة الرئيسيّة أوامر تكوِّن كائنَيْن باسم st1 أو st2 من نمط **Student**، لكليهما اسمان مختلفان ولكليهما نفس سنة الميلاد.

/ يتبع في صفحة 3 /

2. أمامك قطعة من عمليّة رئيسيّة في الفئة Program، مكتوبة بلغة Java وبلغة C# .
تمّ في العمليّة الرئيسيّة تعريف مصفوفة أحاديّة الأبعاد arr من نمط صحيح. المصفوفة تحوي أعدادًا أكبر من 0.

C#

```
for (int i = 0; i < arr.Length; i = i + 2)
{
    if ((arr[i] > 9) && (arr[i] < 100))
    {
        arr[i] = 0;
    }
}
```

Java

```
for (int i = 0; i < arr.length; i = i + 2)
{
    if ((arr[i] > 9) && (arr[i] < 100))
    {
        arr[i] = 0;
    }
}
```

أ. أمامك المصفوفة arr .

	0	1	2	3	4	5	6	7
arr	12	2	324	33	67	888	9	5

تتبّع بواسطة جدول متابعة تنفيذ قطعة العمليّة.
يجب أن يشمل جدول المتابعة:
عمودًا لـ i ، وعمودًا لـ arr[i] ، وعمودًا يُذكَر فيه إذا كان الشرط الذي في الأمر if يتحقّق أم لا يتحقّق.
اعرض المصفوفة بعد تنفيذ القطعة.

ب. أعطِ مثالاً ممثِّلًا لمصفوفة بكِبَر 8 بالنسبة لها، بعد تنفيذ قطعة العمليّة، تكون جميع قيَم المصفوفة لا تساوي 0.

3. اكتب بلغة Java أو بلغة C# عمليّة خارجيّة تتلقّى متغيّرًا بوليانيًّا وعددًا صحيحًا مكوَّنًا من ثلاثة أرقام وأكبر من 0.
إذا كانت قيمة المتغيّر البوليانيّ true – تُعيد العمليّة مجموع أرقام العدد .
خلاف ذلك – تُعيد العمليّة حاصل ضرب أرقام العدد.

/ يتبع في صفحة 4 /

4. معطاة الفئة **Worker** التي تمثّل عاملاً. للفئة صفتان: الاسم – name،
وقيمة الراتب – salary.
تمّ في الفئة تعريف عمليّة بنائيّة عنوانها:
بلغة Java: public Worker(String name, int salary)
بلغة C#: public Worker(string name, int salary)

كما تمّ في الفئة تعريف عمليّة عنوانها:
بلغة Java: public boolean isBig()
بلغة C#: public bool IsBig()
تُعيد العمليّة true – إذا كان الراتب أعلى من 9000 شيكل،
خلاف ذلك – تُعيد العمليّة false.

لكلّ صفة تمّ تعريف عمليّات get وَ set بلغة Java، وعمليّات Get وَ Set بلغة C#.

أمامك قطعة من عمليّة رئيسيّة في الفئة Program، مكتوبة بلغة Java وبلغة C#.

Java

```
Worker w1 = new Worker("Moshe", 10800);
Worker w2 = new Worker("Haim", 4800);
if (w1.isBig() && w2.isBig())
    System.out.println("Happy");
else
    {
        System.out.println("Unhappy");
        if (!w1.isBig())
            System.out.println(w1.getName());
        if (!w2.isBig())
            System.out.println(w2.getName());
    }
```

/ يتبع في صفحة 5 /

C#

```
Worker w1 = new Worker("Moshe", 10800);
Worker w2 = new Worker("Haim", 4800);
if (w1.IsBig() && w2.IsBig())
    Console.WriteLine("Happy");
else
{
    Console.WriteLine("Unhappy");
    if (!w1.IsBig())
        Console.WriteLine(w1.GetName());
    if (!w2.IsBig())
        Console.WriteLine(w2.GetName());
}
```

تتبّع قطعة العمليّة واكتب ماذا يكون المُخرَج. في المتابعة اعرض الكائنات وقيَم صفاتها.
اكتب بجانب كلّ كائن ماذا تُعيد بالنسبة له العمليّة isBig بلغة Java أو IsBig بلغة C# .

5. تقرّر في مدينة معيّنة أنّه يجب أن تكون لجنة للبناية في كلّ بناية فيها أكثر من 4 شقق.
تمّ تعريف الفئة بناية – **Building** التي إحدى صفاتها هي عدد الشقق في البناية:

```
private int numAps;
```

أ. اكتب بلغة Java أو بلغة C# عمليّة في الفئة **Building** تُعيد true – إذا كان يجب أن تكون لجنة للبناية، خلاف ذلك – تُعيد العمليّة false.

ب. في الفئة Test في العمليّة الرئيسيّة تمّ تعريف مصفوفة أحاديّة الأبعاد bdng من نمط **Building**، كلّ واحد من حدودها يمثّل بناية في حيّ معيّن في المدينة.
اكتب بلغة Java أو بلغة C# في العمليّة الرئيسيّة، قطعة برنامج تَعُدُّ وتعرض كمُخرَج عدد البنايات في الحيّ التي يجب أن يكون فيها لجنة للبناية.
عليك استعمال العمليّة التي كتبتَها في البند "أ".

/ يتبع في صفحة 6 /

الفصل الثاني (30 درجة)

أجب عن اثنين من الأسئلة 6-8 (لكلّ سؤال – 15 درجة).

6. تقرّر تنظيم استطلاع للرأي في 248 بلدة، من أجل فحص مستوى رضا السكّان عن أداء المدارس في هذه البلدات. تشترك البلدة في الاستطلاع إذا كانت ملائمة للاستطلاع وفقًا لعدّة شروط. الشروط تتحدّد كلّ سنة من جديد.

معطاة الفئة بلدة – **City** المكتوبة بلغة Java وبلغة C# .

Java

```
public class City
{
    private String name; // اسم البلدة
    private int popul; // عدد سكّان البلدة
    private int numOfSchools; // عدد المدارس في البلدة
    public City(String name, int popul, int numOfSchools) {...} // عمليّة بنائيّة
    public boolean isFit(){...} // عمليّة تُعيد true إذا كانت البلدة ملائمة للاستطلاع وَ false – خلاف ذلك
}
```

C#

```
public class City
{
    private string name; // اسم البلدة
    private int popul; // عدد سكّان البلدة
    private int numOfSchools; // عدد المدارس في البلدة
    public City(string name, int popul, int numOfSchools) {...} // عمليّة بنائيّة
    public bool IsFit(){...} // عمليّة تُعيد true إذا كانت البلدة ملائمة للاستطلاع وَ false – خلاف ذلك
}
```

افترض أنّه تمّ لكلّ صفة تعريف عمليّات get وَ set بلغة Java ، وعمليّات Get وَ Set بلغة C# .

أ. اكتب بلغة Java أو بلغة C# في الفئة Program عمليّة رئيسيّة تتلقّى بالنسبة لكلّ واحدة من البلدات الـ 248 ، التفاصيل التالية:

- اسم البلدة
- عدد سكّان البلدة
- عدد المدارس في البلدة

العمليّة الرئيسيّة تُكوِّن لكلّ بلدة كائنًا من نمط **City**.

العمليّة الرئيسيّة تعرض كمُخرَج:

– لكلّ بلدة: اسم البلدة، وإذا كانت ملائمة للاستطلاع أم غير ملائمة للاستطلاع.

– عدد البلدات التي ليست ملائمة للاستطلاع.

/ يتبع في صفحة 7 /

ب. في سنة 2015 تحدّدت الشروط التالية:

تشترك في الاستطلاع فقط بلدات فيها أكثر من 3 مدارس، وأكثر من 5,000 نسمة.

طبِّق بلغة Java العمليّة isFit أو بلغة C# العمليّة IsFit ، وفقًا للشروط التي تحدّدت في سنة 2015.

/يتبع في صفحة 8/

7. أمامك جزء من واجهة تطبيق الفئة قلم – **Pen**. للفئة أربع صفات:
لون القلم – color من نمط نصّ؛ اسم المنتِج – made من نمط نصّ؛
سعر القلم – price من نمط حقيقيّ؛ وزن القلم – weight من نمط حقيقيّ.

عنوان العمليّة بلغة Java	وصف العمليّة
public Pen(String made)	عمليّة تبني قلمًا لونه "red"، للمنتج made ، وسعره 10.0 شواكل، ووزنه 25.0 غرامًا.
public Pen(String color, String made, double price, double weight)	عمليّة تبني قلمًا لونه color ، للمنتِج made ، وسعره price ، ووزنه weight.
public String getMade()	عمليّة تُعيد اسم منتِج القلم.
public double getPrice()	عمليّة تُعيد سعر القلم.
public void setPrice(double x)	عمليّة تُحدِّث سعر القلم بحيث يكون x .
public boolean isSameMade(Pen other)	عمليّة تُعيد true إذا كان اسم منتِج القلم other مطابقًا لاسم منتِج القلم الحاليّ، خلاف ذلك – تُعيد false .
public boolean isSamePrice(Pen other)	عمليّة تُعيد true إذا كان سعر القلم other مساويًا لسعر القلم الحاليّ، خلاف ذلك – تُعيد false .

عنوان العمليّة بلغة C#	وصف العمليّة
public Pen(string made)	عمليّة تبني قلمًا لونه "red"، للمنتج made ، وسعره 10.0 شواكل، ووزنه 25.0 غرامًا.
public Pen(string color, string made, double price, double weight)	عمليّة تبني قلمًا لونه color ، للمنتِج made ، وسعره price ، ووزنه weight.
public string GetMade()	عمليّة تُعيد اسم منتِج القلم.
public double GetPrice()	عمليّة تُعيد سعر القلم.
public void SetPrice(double x)	عمليّة تُحدِّث سعر القلم بحيث يكون x .
public bool IsSameMade(Pen other)	عمليّة تُعيد true إذا كان اسم منتِج القلم other مطابقًا لاسم منتِج القلم الحاليّ، خلاف ذلك – تُعيد false .
public bool IsSamePrice(Pen other)	عمليّة تُعيد true إذا كان سعر القلم other مساويًا لسعر القلم الحاليّ، خلاف ذلك – تُعيد false .

/يتبع في صفحة 9/

أ. طبِّق في الفئة **Pen**:
بلغة Java العمليّة البنائيّة Pen(String made)
أو بلغة C# العمليّة البنائيّة Pen(string made)

ب. طبِّق في الفئة **Pen** ، بلغة Java العمليّة setPrice أو بلغة C# العمليّة SetPrice.

جـ. طبِّق في الفئة **Pen** ، بلغة Java العمليّة isSamePrice أو بلغة C# العمليّة IsSamePrice.

د. أمامك قطعة من عمليّة رئيسيّة في الفئة TestProg، مكتوبة بلغة Java وبلغة C# .
تتبّع تنفيذ قطعة العمليّة واكتب المُخرَج الذي ينتج.
في المتابعة اعرض الكائنات وقيَم صفاتها.
بجانب كلّ كائن اكتب أسماء العمليّات التي تمّ تفعيلها فيه، وبالنسبة لكلّ عمليّة – القيمة التي أُعيدت.

Java

```
Pen a1 = new Pen("red" , "aaa" , 10.0 , 5.5);
Pen a2 = new Pen("bbb");
Pen a3 = new Pen("black" , "ccc" , 8.0 , 12.5);
a1.setPrice(15.0);
if(!(a1.isSamePrice(a3)))
        if(a2.isSameMade(a1))
            System.out.println("YES");
        else
            System.out.println("NO");
else
        System.out.println("OK");
```

C#

```
Pen a1 = new Pen("red" , "aaa" , 10.0 , 5.5);
Pen a2 = new Pen("bbb");
Pen a3 = new Pen("black" , "ccc" , 8.0 , 12.5);
a1.SetPrice(15.0);
if(!(a1.IsSamePrice(a3)))
        if(a2.IsSameMade(a1))
            Console.WriteLine("YES");
        else
            Console.WriteLine("NO");
else
        Console.WriteLine("OK");
```

/يتبع في صفحة 10 /

8. تنظّم شركة الرحلات "رحلة ممتعة" رحلات للعائلات. يجب على العائلات التي تشترك في الرحلة أن تدفع مقابل الإرشاد ومقابل الوجبة التي تُقدَّم خلال الرحلة.

سعر الإرشاد ثابت – 100 شيكل لكلّ عائلة.

سعر الوجبة في الرحلة هو:

30 شيكلاً للمشترك – للعائلة التي فيها 3 مشتركين أو أقلّ.

28 شيكلاً للمشترك – للعائلة التي فيها 4-5 مشتركين.

26 شيكلاً للمشترك – للعائلة التي فيها أكثر من 5 مشتركين.

أمامك الفئة عائلة – **Family** المكتوبة بلغة Java وبلغة C#.

Java

```
public class Family
{
    private int num; // عدد المشتركين من العائلة
    private String familyName; // اسم العائلة
    public Family(int num , String familyName) {...}
}
```

C#

```
public class Family
{
    private int num; // عدد المشتركين من العائلة
    private string familyName; // اسم العائلة
    public Family(int num , string familyName) {...}
}
```

لكلّ صفة تمّ تعريف عمليّات get وَ set بلغة Java ، وعمليّات Get وَ Set بلغة C#.

أ. اكتب في الفئة **Family** ، بلغة Java عمليّة calcPrice أو بلغة C# عمليّة CalcPrice ، تُعيد الثمن الذي على العائلة دفعه مقابل الرحلة.

ب. اكتب بلغة Java أو بلغة C# ، في الفئة Program ، عمليّة رئيسيّة تستقبل بالنسبة لكلّ عائلة تشترك في الرحلة عدد المشتركين فيها واسم العائلة. ينتهي الإدخال عندما يُستقبَل 0 بالنسبة لعدد المشتركين من العائلة.

العمليّة الرئيسيّة تُكوِّن لكلّ عائلة كائنًا من نمط **Family** .

العمليّة الرئيسيّة تعرض كمُخرَج:

- لكلّ عائلة: اسمها والثمن الذي ستدفعه مقابل الرحلة.
- العدد الكلّيّ للمشتركين في الرحلة.
- أعلى ثمن ستدفعه عائلة ما مقابل الرحلة.

عليك استعمال العمليّة التي كتبتَها في البند "أ".

/يتبع في صفحة 11 /

انتبه: تكملة الامتحان في الصفحة التالية.

/يتبع في صفحة 12 /

الفصل الثالث (20 درجة)

أجب عن أحد السؤالين 9-10.

9. معطاة الفئة **Card** التي تمثّل ورقة شَدّة. للفئة صفتان: قيمة value، بين 1 وَ 9، من نمط صحيح؛ لون color من نمط صحيح: 1 يمثّل أحمر، 2 يمثّل أصفر، 3 يمثّل أخضر، 4 يمثّل أزرق.

وصف العمليّة	عنوان العمليّة بلغة Java
عمليّة تبني ورقة شَدّة قيمتها value ولونها color.	public Card(int value, int color)
عمليّة تُعيد true إذا كانت قيمة ورقة الشَّدّة other مساوية لقيمة ورقة الشَّدّة الحاليّة وأيضًا إذا كان لون ورقة الشَّدّة other مطابقًا للون ورقة الشَّدّة الحاليّة. خلاف ذلك – تُعيد العمليّة false.	public boolean isSame(Card other)
عمليّة تُعيد: 1 — إذا كانت قيمة ورقة الشَّدّة الحاليّة أكبر من قيمة ورقة الشَّدّة other، 2 — إذا كانت قيمة ورقة الشَّدّة الحاليّة أصغر من قيمة ورقة الشَّدّة other، 0 — إذا كانت قيمتا ورقة الشَّدّة الحاليّة وورقة الشَّدّة other متساويتين.	public int compareVal(Card other)

وصف العمليّة	عنوان العمليّة بلغة C#
عمليّة تبني ورقة شَدّة قيمتها value ولونها color.	public Card(int value, int color)
عمليّة تُعيد true إذا كانت قيمة ورقة الشَّدّة other مساوية لقيمة ورقة الشَّدّة الحاليّة وأيضًا إذا كان لون ورقة الشَّدّة other مطابقًا للون ورقة الشَّدّة الحاليّة. خلاف ذلك – تُعيد العمليّة false.	public bool IsSame(Card other)
عمليّة تُعيد: 1 — إذا كانت قيمة ورقة الشَّدّة الحاليّة أكبر من قيمة ورقة الشَّدّة other، 2 — إذا كانت قيمة ورقة الشَّدّة الحاليّة أصغر من قيمة ورقة الشَّدّة other، 0 — إذا كانت قيمتا ورقة الشَّدّة الحاليّة وورقة الشَّدّة other متساويتين.	public int CompareVal(Card other)

/يتبع في صفحة 13/

معطاة الفئة علبة أوراق شَدّة – **Deck** التي لها صفة واحدة – مصفوفة أحاديّة الأبعاد allCards من نمط **Card**.

وصف العمليّة	عنوان العمليّة بلغة Java
عمليّة تبني علبة أوراق شَدّة: العمليّة تعرّف مصفوفة allCards بكبَر 36 فيها 36 ورقة شَدّة تختلف عن بعضها البعضَ.	public Deck()
عمليّة تُعيد ورقة الشَّدّة الموجودة في المكان i في المصفوفة allCards.	public Card seeCard(int i)

وصف العمليّة	عنوان العمليّة بلغة C#
عمليّة تبني علبة أوراق شَدّة: العمليّة تعرّف مصفوفة allCards بكبَر 36 فيها 36 ورقة شَدّة تختلف عن بعضها البعضَ.	public Deck()
عمليّة تُعيد ورقة الشَّدّة الموجودة في المكان i في المصفوفة allCards.	public Card SeeCard(int i)

أ. طبِّق بلغة Java العمليّة seeCard أو بلغة C# العمليّة SeeCard التي في الفئة **Deck**.

أمامك وصف للعبة شدّة للاعبَيْن.

في بداية اللعبة يكون بحوزة كلّ واحد من اللاعبَيْن علبة أوراق شدّة.

في كلّ دور:

كلّ واحد من اللاعبَيْن يقترع عددًا عشوائيًّا i بين 0 وَ 35 (بما في ذلك 0 وَ 35) ويعرض ورقة الشَّدّة الموجودة في المكان i في المصفوفة allCards من علبته.

- إذا كانت ورقتا الشَّدّة متطابقتين (في القيمة وفي اللون)، يحصل كلّ واحد من اللاعبَيْن على نقطة واحدة.
- اللاعب الذي قيمة ورقة شَدّته أكبر من قيمة ورقة شَدّة اللاعب الآخر، يحصل على 3 نقاط.

بالإضافة إلى ذلك:

- كلّ لاعب لون ورقة شَدّته أخضر، يحصل على نقطتَيْن.

تنتهي اللعبة عندما يُجمِّع أحد اللاعبَيْن 28 نقطة على الأقلّ.

ب. اكتب بلغة Java أو بلغة C# قطعة من عمليّة رئيسيّة في الفئة Program تحاكي لعبة الشَّدّة.
قطعة البرنامج تطبع عدد النقاط التي جمّعها كلّ واحد من اللاعبَيْن في نهاية اللعبة.
عليك استعمال عمليّات الفئتين **Card** وَ **Deck**، ولستَ مُلزَمًا بتطبيقها.
افترض أنّه قد تمّ في الفئتين **Card** وَ**Deck** تعريف عمليّات get وَ set بلغة Java، وعمليّات Get وَ Set بلغة C# لكلّ صفة.

/يتبع في صفحة 14 /

10. مصفوفة خفيفة – sparse matrix – هي مصفوفة ثنائيّة الأبعاد من نمط صحيح قيمة أكثريّة الحدود فيها هي 0.

على سبيل المثال المصفوفة الثنائيّة الأبعاد التي بِكِبَر 5×4 التي أمامك.

	0	1	2	3	4
0	0	1	0	0	0
1	0	0	0	0	9
2	0	0	0	0	0
3	7	0	0	0	0

من أجل توفير مكان في ذاكرة الحاسوب، يمكن فقط حفظ حدود المصفوفة الثنائيّة الأبعاد التي لا تساوي 0. بالنسبة لكلّ حدّ تُحفَظ قيمته ومكانه في المصفوفة الثنائيّة الأبعاد: رقم السطر ورقم العمود.

بالنسبة للمصفوفة الثنائيّة الأبعاد التي في المثال، 3 الحدود التي لا تساوي 0 تُحفَظ على النحو التالي:

value = 1 row = 0 col = 1	value = 9 row = 1 col = 4	value = 7 row = 3 col = 0

تمّ تعريف الفئة **Item** التي تمثّل حدًّا لا يساوي 0 في المصفوفة الثنائيّة الأبعاد. صفاتها هي: قيمة الحدّ في المصفوفة الثنائيّة الأبعاد، ومكانه: رقم السطر ورقم العمود.

أ. **(1)** اكتب بلغة Java أو بلغة C# عنوان وصفات الفئة **Item**.

(2) اكتب بلغة Java أو بلغة C# عمليّة بنائيّة في الفئة **Item**، تتلقّى كبارامترات قيَمًا لكلّ صفة.

/يتبع في صفحة 15 /

تمّ تعريف فئة إضافيّة – **Sparse**، صفاتها هي :
مصفوفة أحاديّة الأبعاد itemAr من نمط **Item**،
rows – من نمط صحيح، عدد الأسطر في **المصفوفة الخفيفة**،
cols – من نمط صحيح، عدد الأعمدة في **المصفوفة الخفيفة**.

ب. اكتب بلغة Java أو بلغة C# عنوان الفئة **Sparse** وصفاتها.

في الفئة **Sparse** معطاة عمليّة عنوانها :
بلغة Java: public int countNoZero(int[ ][ ] mat)
بلغة C#: public int CountNoZero(int[,] mat)
العمليّة تتلقّى مصفوفة ثنائيّة الأبعاد وتُعيد عدد الحدود التي لا تساوي 0 فيها.

جـ. اكتب بلغة Java أو بلغة C# عمليّة بنائيّة في الفئة **Sparse** ، تتلقّى كبارامتر مصفوفة ثنائيّة الأبعاد من نمط صحيح هي **مصفوفة خفيفة**.
عليك استعمال العمليّة المعطاة.

ב ה צ ל ח ה!
نتمنّى لك النّجاح!