

(في الامتحان فصلان. يجب الطالب عن أربعة أسئلة، ولكل سؤال 27 علامة. أسئلة الفصل الأول تُكتب بلغة Java أو بلغة C#. والطالب يكتب بلغة واحدة فقط. هنا نكتب الحل باللغتين: Java أو C# ثم C#).

الفصل الأول

1. معطاة الفئة **DigitCount**، ولها صفتان :

الصفة	الشرح
private int digit;	قيمة الرقم (0-9)
private int count;	عدد مرّات ظهور الرقم

افتراضوا أنّه توجد عمليّات get/Get و set/Set لصفتيّ الفئة.

كما توجد في الفئة العمليّة التالية :

العملية	الشرح
<pre>public DigitCount (int digit, int count) { this.digit = digit; this.count = count; }</pre>	عملية بناءية

أ. طبّقوا العمليّة التي أمامكم :

Java – public static Node<DigitCount> buildList (int num)

C# – public static Node<DigitCount> BuildList (int num)

تتلقّى العمليّة عددًا صحيحًا موجبًا num أكبر من 0 ، وتعيد قائمة مرتبطة من نوع DigitCount.

بالنسبة لكل رقم (0-9) في العدد num ، يُحفظ في القائمة كائن من الفئة DigitCount الذي قيمته الرقم نفسه وعدد مرّات ظهوره في العدد num. تُركّب القائمة حسب القواعد التالية :

- تكون القائمة مرتّبة حسب قيمة الأرقام بترتيب تنازليّ (من الرقم ذي القيمة الأعلى إلى الرقم ذي القيمة الأقلّ).
- كل رقم يظهر مرّة واحدة فقط في القائمة.
- فقط الأرقام التي تظهر في العدد num تظهر في القائمة.

مثال: بالنسبة لاستدعاء العملية مع $num = 3389920$ ، وأيضًا بالنسبة لـ $num = 9389302$ ، تُعاد القائمة:



الشرح: توجد في القائمة حلقة بالنسبة لكل رقم يظهر في العدد num . الحلقة تُمثل الرقم وعدد مرّات ظهور الرقم في num . القائمة تحوي فقط الأرقام الموجودة في num ، وهي مرتّبة حسب قيمة الأرقام من أكبر قيمة إلى أصغر قيمة.

ب. طبّقوا العملية التي أمامكم:

Java: public static int buildMaxNum (int num)

C#: public static int BuildMaxNum (int num)

تتلقّى العملية عددًا صحيحًا num أكبر من 0 ، وتُعيد أكبر عدد يمكن تكوينه بواسطة الأرقام في العدد num ، حسب عدد مرّات ظهور كل رقم. ملاحظة: يمكن استعمال العملية التي كتبتموها في البند "أ".

مثال: بالنسبة لـ $num = 3389920$ وأيضًا بالنسبة لـ $num = 9389302$ ، تُعيد العملية العدد: 9983320. الشرح: هذا العدد هو العدد الأكبر الذي يمكن تكوينه بواسطة تغيير ترتيب الأرقام في العدد num .

الحل: أ. منظومة عدّادات ثم قائمة من 9 إلى 0 ب. نمّر على القائمة ونلصق الأرقام

(الحلّ والشرح: فيفو · vevo. لا تنشر وزارة التربية حلولًا لهذا الامتحان. لا يوجد مترجم Java ولا C# على جهازنا، لذلك ترجمنا الكود سطرًا سطرًا إلى Python وشغلناه فعليًا على مثالي الورقة وعلى حالات حدّية وعلى عشوائيين، وقارنًا بالجواب الصحيح).

الحلّ الكامل:

البند أ

الفكرة: نمّر أولًا على منازل العدد ونعدّ كم مرّة يظهر كل رقم. نحفظ العدّ في منظومة counts من 10 خانات: الخانة counts[d] تحفظ عدد مرّات ظهور الرقم d. نأخذ المنزلّة الأخيرة بـ $num \% 10$ ، ثم نحذفها بـ $num / 10$ ، حتّى يصير العدد صفرًا.

num = 3389920

counts	1	0	1	2	0	0	0	0	1	2
index	0	1	2	3	4	5	6	7	8	9

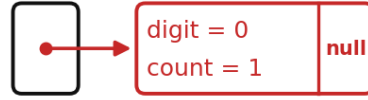
المنظومة counts بعد المرور على منازل العدد 3389920: الخانات الملوّنة هي الأرقام الموجودة في العدد.

الآن نبني القائمة. نمّر على الأرقام من 0 إلى 9. كل رقم عدّه أكبر من صفر نضعه في حلقة جديدة في رأس القائمة. لأنّ الرقم الأكبر يدخل أخيرًا، يكون هو في الرأس، فتخرج القائمة مرتّبة تنازليًا بلا أيّ فرز. كل رقم

يدخل مرّة واحدة فقط، والأرقام التي عدّها صفر لا تدخل أبدًا.

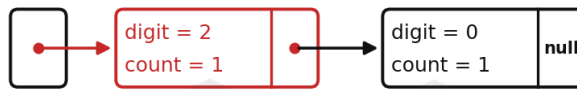
الخطوة 1: أوّل رقم موجود هو $d = 0$ وعدّه 1. القائمة كانت فارغة ($head = null$)، فتصير الحلقة الجديدة (بالأحمر) هي القائمة كلّها.

head



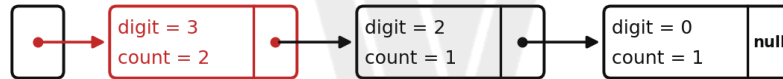
الخطوة 2: الرقم $d = 2$ عدّه 1. ننشئ حلقة جديدة (بالأحمر) تشير إلى الرأس القديم، وتصير هي الرأس.

head



الخطوة 3: الرقم $d = 3$ عدّه 2. ننشئ حلقة جديدة (بالأحمر) تشير إلى الرأس القديم، وتصير هي الرأس.

head



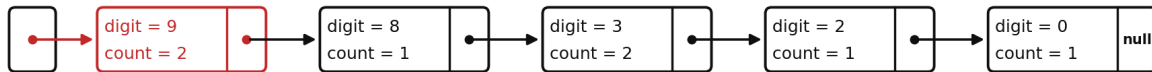
الخطوة 4: الرقم $d = 8$ عدّه 1. ننشئ حلقة جديدة (بالأحمر) تشير إلى الرأس القديم، وتصير هي الرأس.

head



الخطوة 5: الرقم $d = 9$ عدّه 2. ننشئ حلقة جديدة (بالأحمر) تشير إلى الرأس القديم، وتصير هي الرأس.

head



(القائمة بعد الخطوة 5 هي نفسها القائمة في مثال الورقة. للعدد 9389302 نحصل على المنظومة نفسها، فتخرج القائمة نفسها).

الكود بلغة Java. السطر 4 يعدّ المنزلّة الأخيرة، والسطر 10 والسطر 11 ينشئان الحلقة ويضعانها في رأس القائمة:

Java

```
1 public static Node<DigitCount> buildList(int num) {
2     int[] counts = new int[10];           // counts[d] = times d appears
3     while (num > 0) {
4         counts[num % 10]++;                // count the last digit
5         num = num / 10;                   // drop the last digit
6     }
7     Node<DigitCount> head = null;
8     for (int d = 0; d <= 9; d++) {
9         if (counts[d] > 0) {
10            DigitCount dc = new DigitCount(d, counts[d]);
11            head = new Node<DigitCount>(dc, head); // add at front
12        }
13    }
14    return head;
15 }
```

الكود بلغة C# (المنطق نفسه؛ هنا السطر 6 يعدّ المنزلّة، والسطر 14 والسطر 15 يضيفان الحلقة في الرأس):

C#

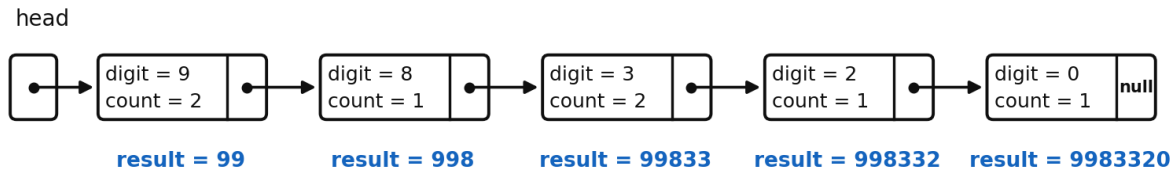
```
1 public static Node<DigitCount> BuildList(int num)
2 {
3     int[] counts = new int[10];           // counts[d] = times d appears
4     while (num > 0)
5     {
6         counts[num % 10]++;                // count the last digit
7         num = num / 10;                   // drop the last digit
8     }
9     Node<DigitCount> head = null;
10    for (int d = 0; d <= 9; d++)
11    {
12        if (counts[d] > 0)
13        {
14            DigitCount dc = new DigitCount(d, counts[d]);
15            head = new Node<DigitCount>(dc, head); // add at front
16        }
17    }
18    return head;
19 }
```

(حالات حدّيّة شغلناها: العدد 7 يعطي حلقة واحدة. العدد 1000 يعطي حلقتين: الرقم 1 مرّة واحدة، ثمّ الرقم 0 ثلاث مرّات. العدد 111 يعطي حلقة واحدة عدّها 3.)

head = new Node<DigitCount>(dc, head);

الجواب النهائي للبند أ (قلب الحلّ)

الفكرة: العدد الأكبر يبدأ بالأرقام الكبيرة. القائمة من البند أ مرتبة من 9 إلى 0 أصلاً. لذلك نمرّ عليها من الرأس، ونلصق كل رقم في آخر النتيجة بعدد مرّات ظهوره. لصق رقم في آخر عدد يعني: ضرب النتيجة في 10 ونضيف الرقم.



الممرور على القائمة من الرأس: تحت كل حلقة قيمة result بعد لصق الرقم بعدد مرّات ظهوره.

جدول تتبّع للعدد 3389920 (كل سطر هو دورة واحدة من الحلقة الداخلية):

الحلقة (digit, count)	الحساب	result بعد الدورة
9, 2	$0 * 10 + 9$	9
9, 2	$9 * 10 + 9$	99
8, 1	$99 * 10 + 8$	998
3, 2	$998 * 10 + 3$	9983
3, 2	$9983 * 10 + 3$	99833
2, 1	$99833 * 10 + 2$	998332
0, 1	$998332 * 10 + 0$	9983320

الكود بلغة Java. السطر 2 يستدعي العملية من البند أ، والسطر 7 يلصق الرقم في آخر النتيجة:

Java

```

1 public static int buildMaxNum(int num) {
2     Node<DigitCount> pos = buildList(num); // digits from 9 down to 0
3     int result = 0;
4     while (pos != null) {
5         DigitCount dc = pos.getValue();
6         for (int i = 0; i < dc.getCount(); i++) {
7             result = result * 10 + dc.getDigit(); // append digit
8         }
9         pos = pos.getNext();
10    }
11    return result;
12 }
  
```

C#

```
1 public static int BuildMaxNum(int num)
2 {
3     Node<DigitCount> pos = BuildList(num);    // digits from 9 down to 0
4     int result = 0;
5     while (pos != null)
6     {
7         DigitCount dc = pos.GetValue();
8         for (int i = 0; i < dc.GetCount(); i++)
9         {
10            result = result * 10 + dc.GetDigit();    // append digit
11        }
12        pos = pos.GetNext();
13    }
14    return result;
15 }
```

(شغلنا الترجمة على مثالي الورقة فأعادت 9983320 في الحالتين، وعلى 1000 فأعادت 1000، وعلى 10 فأعادت 10. ملاحظة: إذا كان num من عشر منازل، قد يكون العدد الأكبر خارج مدى int، لكن توقيع الورقة يعيد int، فلا نعالج ذلك.)

result = result * 10 + dc.getDigit();

الجواب النهائي للبند ب (للمثال يُعاد 9983320)

نص الأسئلة والصور: وزارة التربية والتعليم، النموذج 899271، صيف 2026. الحل والشرح: فيفو · vevo (لا تنشر وزارة التربية طوّل لهذا الامتحان). حقوق الطبع للوزارة: «النسخ أو النشر ممنوعان إلا بإذن من وزارة التربية والتعليم».

VEVO