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 input = 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 الفئة **Order** – طلبيّة لزبون، لها صفتان:

- id – رقم هويّة الزبون، من نمط صحيح.
- count – كمّيّة المنتَجات التي طُلبت، من نمط صحيح.

افترِضوا أنّه توجد عمليّات get/Get وَ set/Set لصفتَي الفئة، وعمليّة بنائيّة تتلقّى قيَمًا بالنسبة لصفتَي الفئة.

في شركة الشحن "برق" بُنِيَ دَوْر qOrder – من نمط **Order**، يحفظ الطلبيّات المختلفة للزبائن في يوم معيّن، التي يجب على شركة الشحن أن تزوّدها.

ملاحظتان:

- يمكن أن يكون في الدَّوْر عدّة طلبيّات لنفس الزبون – id (إذا كان للزبون أكثر من طلبيّة واحدة في نفس اليوم).
- مكان الطلبيّات في الدَّوْر هو ليس حسب ترتيب معيّن (كذلك طلبيّات نفس الزبون يمكن أن تظهر في أماكن مختلفة في الدَّوْر).

في نهاية اليوم، من أجل زيادة نجاعة الشحن، يُجرون، في دَوْر جديد، توحيدًا للطلبيّات حسب رقم هويّة الزبون (id)، بحيث تُحفَظ لكلّ زبون في الدَّوْر طلبيّة واحدة فقط، مع العدد الكلّيّ لجميع المنتَجات التي طلبها في جميع الطلبيّات (count). هكذا مثلًا، إذا كان في الدَّوْر qOrder 3 طلبيّات لنفس الزبون: طلبيّة لِـ 20 منتَجًا وطلبيّة لِـ 15 منتَجًا وطلبيّة لِـ 30 منتَجًا، بعد التوحيد، يظهر الزبون مرّة واحدة فقط في الدَّوْر الجديد – مع طلبيّة لِـ 65 منتَجًا.

أ. **(1)** طبِّقوا العمليّة التي أمامكم:

Java – public static Queue<Order> uniteOrders (Queue<Order> qOrder)

C# – public static Queue<Order> UniteOrders (Queue<Order> qOrder)

تتلقّى العمليّة دَوْرًا فيه طلبيّات – qOrder، يمكن أن يظهر فيه نفس الزبون أكثر من مرّة واحدة، وتُعيد دورًا جديدًا فيه توحيد الطلبيّات كما شُرِح أعلاه.

ملاحظات:

- لا توجد أهمّيّة لترتيب الطلبيّات بعد توحيد الطلبيّات.
- لا تستعمِلوا في هذا البند مصفوفة أو قائمة مرتبطة (أو أيّ مبنى بيانات آخر باستثناء الدَّوْر). الحلّ الذي يتضمّن استعمالها لن يحصل على درجات.
- يمكن تغيير الدَّوْر المتلقَّى.

(2) ما هي تعقيدات زمن تشغيل العمليّة؟ علِّلوا إجابتكم.

(انتبهوا: تكملة السؤال في الصفحة التالية.)

/ يتبع في صفحة 3 /

ب. **(1)** "زبون عاديّ" هو الزبون الذي يطلب **بالمجمل** أقلّ من 10 منتَجات في يوم معيّن، وَ"زبون مفضَّل" هو الزبون الذي يطلب **بالمجمل** 10 منتَجات أو أكثر في يوم معيّن.

طبِّقوا العمليّة التي أمامكم:

Java – public static Queue<Integer> preferredClients (Queue<Order> qOrder)

C# – public static Queue<int> PreferredClients (Queue<Order> qOrder)

تتلقّى العمليّة دَوْر طلبيّات في يوم معيّن – qOrder من نمط **Order** (دَوْرًا "ليس موحَّدًا"، يمكن أن يظهر فيه نفس الزبون عدّة مرّات)، وتُعيد دَوْرًا من نمط **صحيح**. في بداية الدَّوْر المُعاد، تظهر أرقام هويّات (id) كلّ مَن هو "زبون مفضَّل" وبعدها تظهر أرقام هويّات (id) كلّ مَن هو "زبون عاديّ" (بدون أهمّيّة لترتيب الزبائن في كلّ مجموعة).

بإمكانكم استعمال العمليّة التي كتبتموها في البند "أ".

ملاحظات:

- كلّ زبون (id) يظهر مرّة واحدة فقط في الدَّوْر المُعاد.
- لا تستعمِلوا في هذا البند مصفوفة أو قائمة مرتبطة (أو أيّ مبنى بيانات آخر باستثناء الدَّوْر). الحلّ الذي يتضمّن استعمالها لن يحصل على درجات.
- يمكن تغيير الدَّوْر المتلقّى.

(2) ما هي تعقيدات زمن تشغيل العمليّة؟ علِّلوا إجابتكم.

/ يتبع في صفحة 4 /