Государство Израиль
Министерство просвещения
Тип экзамена: на аттестат зрелости
Время проведения экзамена: лето 2024 года
Номер вопросника: 899381

מדינת ישראל
משרד החינוך
סוג הבחינה: בגרות
מועד הבחינה: קיץ תשפ"ד, 2024
מספר השאלון: 899381

Обратите внимание: в этом вопроснике есть специальные инструкции.
Отвечайте на вопросы, следуя этим инструкциям.

שימו לב: בבחינה זו יש הנחיות מיוחדות.
יש לענות על השאלות על פי הנחיות אלה.

Компьютеры

מ ד ע י ה מ ח ש ב

Указания

הוראות

а. Продолжительность экзамена: три с половиной часа.

א. משך הבחינה: שלוש שעות וחצי.

б. Строение вопросника и ключ к оценке:
в этом вопроснике три раздела.

Раздел первый:	$(1 \times 10) + (1 \times 15)$	–	25 баллов
Раздел второй:	(3×25)	–	75 баллов
Раздел третий:			
Всего		–	100 баллов

Обратите внимание: если вы выбрали отвечать на вопросы третьего раздела, выбирайте вопросы только из одного курса.

ב. מבנה השאלון ומפתח ההערכה:
בשאלון זה שלושה פרקים.

פרק ראשון	$(10\times1) + (15\times1)$	25 נקודות
פרק שני	(25×3)	75 נקודות
פרק שלישי		
סך הכול		100 נקודות

שימו לב: אם תבחרו לענות על שאלות מן הפרק השלישי: בחרו בשאלות מתוך מסלול אחד בלבד.

в. Разрешенный вспомогательный материал:
любой вспомогательный материал, за исключением калькулятора с возможностью программирования.

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

г. Особые указания:

(1) Если вы решите отвечать на вопросы **только** первого и второго разделов, **напишите на внешней обложке** тетради **"ללא מסלול"**, иначе – напишите **название курса**, **который вы изучали**. Курс – это один из трех следующих курсов: Алгоритмы, Вычислительные модели, Объектно-ориентированное программирование.

(2) Все программы, которые вы должны написать на языке программирования в первом и втором разделах, пишите только на одном языке – Java или C# .

Примечание: баллы сниматься не будут, если в программах вы напишете большую букву вместо маленькой или наоборот.

ד. הוראות מיוחדות:

(1) אם תבחרו לענות על שאלות **רק** מן הפרק הראשון והפרק השני, **רשמו על הכריכה החיצונית** של המחברת **ללא מסלול**, אחרת רשמו את **שם המסלול שלמדתם**.
המסלול הוא אחד משלושת המסלולים האלה: אלגוריתמים, מודלים חישוביים, תכנות מונחה עצמים.

(2) את כל התוכניות שיש לכתוב בשפת מחשב בפרקים הראשון והשני כִּתבו בשפה אחת בלבד – Java או C# .

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

יש לכתוב במחברת הבחינה בלבד. יש לרשום "טיוטה" בראש כל עמוד המשמש טיוטה.
כתיבת טיוטה בדפים שאינם במחברת הבחינה עלולה לגרום לפסילת הבחינה.
Пишите только в экзаменационной тетради. Напишите слово «טיוטה» в начале каждой страницы, отведенной вами под черновик. Выполнение любых черновых записей на листах, не относящихся к экзаменационной тетради, может привести к тому, что экзамен будет аннулирован.

Желаем успеха!

ב ה צ ל ח ה !

В о п р о с ы

Примечание: в каждом вопросе, в котором требуется ввод данных, нет необходимости проверять их корректность.

Для решающих на языке Java: в каждом вопросе, в котором требуется ввод данных, считайте, что в программе написана команда:

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

РАЗДЕЛ ПЕРВЫЙ (25 баллов)

Ответьте на вопрос 1 – обязательный (10 баллов).

1. Напишите внешний метод [פעולה חיצונית], с названием biggestSum на языке Java или с названием BiggestSum на языке C# , который получает массив целых чисел arr , в котором нет отрицательных чисел (есть числа, которые больше или равны 0). Число 0 появляется в массиве по меньшей мере два раза.

Метод возвращает самую большую из сумм чисел, расположенных между каждыми двумя числами 0 в массиве.

Предположите, что между каждыми двумя числами 0 в массиве есть положительные числа.

Пример:

Для массива, приведенного ниже, метод вернет 17.

	0	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*
arr	33	0	**5**	**4**	0	**17**	0	**4**	**10**	0	5	14

9 17 14

Объяснение: 17 – это самая большая сумма из полученных сумм чисел между каждыми двумя числами 0 в массиве (9 ,17 ,14) .

/продолжение на странице 3/

Ответьте на один из вопросов 2–3 (15 баллов).

2. После выборов секретарь Кнессета начал распределять членов Кнессета по различным парламентским комиссиям.

Для этого были определены классы: **Member** – член Кнессета и **Committee** – комиссия.

В классе Member два атрибута [תכונות]:

- name – имя члена Кнессета (считайте, что нет двух членов Кнессета с одинаковыми именами).
- isCoal – булево значение: true, если член Кнессета входит в коалицию, иначе – false .

В классе Committee три атрибута [תכונות]:

- name – название комиссии
- members – массив типа Member, включающий членов Кнессета, распределенных в комиссию
- count – число работающих членов комиссии (данные членов комиссии хранятся последовательно с начала массива).

Предположите, что определены методы get/Get и set/Set для каждого из атрибутов этого класса.

(א) Напишите названия классов Member и Committee и их атрибуты.

(ב) Напишите внутренний метод [פעולה פנימית] класса Committee с названием total на языке Java или с названием Total на языке C# , получающий параметр belong , булевого типа.
Если belong равен true , метод возвращает число членов комиссии, входящих в коалицию, иначе (если belong равен false) метод возвращает число членов комиссии, не входящих в коалицию.

При распределении членов Кнессета в комиссии секретарь Кнессета соблюдает два следующих правила:

– в каждой комиссии число членов Кнессета, входящих в коалицию, будет больше, чем число членов Кнессета, не входящих в коалицию.
– в каждой комиссии число членов Кнессета будет меньше 16 .

(ג) Реализуйте следующий внешний метод [פעולה חיצונית]:

Java: public static int amount (Committee [] arr, Member m)

C#: public static int Amount (Committee [] arr, Member m)

Метод получает полный массив arr всех комиссий Кнессета и определенного члена Кнессета m , который еще не распределен ни в одну комиссию.

Метод вернет количество комиссий, в которые можно распределить этого члена Кнессета (в соответствии с правилами секретаря Кнессета).

Необходимо использовать метод, который вы написали в пункте (ב).

/продолжение на странице 4/

3. "Массив станков для производства болтов" – это целочисленный массив, в каждой ячейке которого находится число **секунд**, которые требуются для одного определенного станка, чтобы произвести **один** болт (чем меньше число секунд, тем больше скорость, с которой станок производит болты). Массив не отсортирован.
Пример: в следующем массиве станку с индексом 0 требуется только одна секунда, чтобы произвести один болт, станку с индексом 1 требуется 9 секунд, чтобы произвести один болт, станку с индексом 2 требуется 3 секунды, чтобы произвести один болт.

	0	*1*	*2*
"Массив станков для производства болтов"	1	9	3

На фабрике по производству болтов "Болтовня" есть несколько станков для производства болтов. Фабрика хранит в "массиве станков для производства болтов" arr число секунд, которое требуется для каждого из ее станков, чтобы произвести один болт. Фабрика использует все станки для производства болтов одновременно.

(א) Напишите внешний метод [פעולה חיצונית] с названием total на языке Java или с названием Total на языке C# , получающий массив arr и целое число секунд numSecond .
Метод возвращает общее число болтов, которое производит фабрика на всех своих станках в течение numSecond .
Например: для массива из примера выше и numSeconds = 5 метод вернет 6 .
Объяснение: производительность станков за промежуток времени 5 секунд описана ниже:
станок с индексом 0 производит пять болтов (за каждую секунду один болт),
станок с индексом 1 не производит ни одного болта (потому что ему требуется 9 секунд, чтобы произвести один болт),
а станок с индексом 2 производит один болт.
Всего: шесть болтов (5 + 1).

(ב) Напишите внешний метод с названием minTime на языке Java или с названием MinTime на языке C# , получающий массив arr и количество болтов, целое число, amount .
Метод возвращает минимальное число секунд, требуемое для того, чтобы фабрика смогла произвести по меньшей мере amount болтов (на всех станках вместе).
Можно использовать метод, который вы написали в пункте (א).
Например: для массива из примера выше и amount = 7 метод вернет 6 .
Объяснение: производительность станков за промежуток времени 6 секунд описана ниже:
станок с индексом 0 производит шесть болтов,
станок с индексом 1 не производит ни одного болта,
а станок с индексом 2 производит два болта.
Всего: восемь болтов (6 + 2).
Иными словами, трем станкам требуется 6 секунд, чтобы произвести по меньшей мере семь болтов в общей сложности. За менее чем 6 секунд станки не успевают произвести семь болтов (например, за промежуток времени в 5 секунд три станка производят вместе всего шесть болтов, согласно описанию в примере пункта (א)).

/продолжение на странице 5/

Вы должны ответить в общей сложности на три вопроса из второго и третьего разделов (за каждый вопрос – 25 баллов).

РАЗДЕЛ ВТОРОЙ

Внимание: во всех вопросах, где требуется реализация [מימוש], разрешается пользоваться методами классов Очередь, Стек, Двоичное дерево и Звено [תור, מחסנית, עץ בינרי וחוליה], не реализуя их. Если вы используете дополнительные методы, вы должны их реализовать.

4. "Магический элемент" [איבר קסם] – это элемент очереди чисел, значение которого равно сумме значений предыдущего и последующего элементов.

Примечание: первое число в очереди и последнее число в очереди не являются "магическими элементами".

(א) Напишите метод [פעולה] с заголовком isMagic на языке Java или IsMagic на языке C# , принимающий очередь q с элементами целочисленного типа и целое число m , которое больше 0 и меньше размера очереди или равно ему.

Метод возвращает true , если элемент на месте m в очереди является "магическим элементом", иначе метод возвращает false.

Примечания: – По окончании работы метода необходимо сохранить строение очереди в том виде, в котором она была получена.

– В этом вопросе **запрещено** пользоваться массивом или связным списком [רשימה מקושרת]. Решение, в котором они используются, не будет засчитано.

Пример: для приведенной ниже очереди:

	Начало очереди						Конец очереди	
	1	*2*	*3*	*4*	*5*	*6*	*7*	
	5	11	6	9	3	6	3	

для m = 1 метод возвращает false (первое число в очереди – это не "магический элемент")

для m = 2 метод возвращает true (5 + 6 = 11)

для m = 3 метод возвращает false (11 + 9 ≠ 6)

(ב) Напишите метод [פעולה] с названием nMagic на языке Java или NMagic на языке C# , принимающий очередь q с элементами целочисленного типа и целое число n , которое больше 0 и меньше размера очереди или равно ему.

Метод возвращает true , если все элементы, которые находятся на местах, кратных n (место n в очереди, место 2n в очереди и так далее с шагом n мест) – "магические элементы". Иначе метод возвращает false .

Можно использовать метод, который вы написали в пункте (א).

Примечания: – В этом методе не нужно сохранять полученную очередь.

– В этом пункте **не разрешается** использовать массив или связный список. Решение, в котором они используются, не будет засчитано.

Примеры: для очереди из приведенного выше примера:

Для n = 2 метод вернет true , потому что все элементы, которые находятся на местах, кратных 2 (2 , 4 , 6) – это "магические элементы".

Для n = 4 метод вернет true , потому что элемент на месте 4 – это "магический элемент" (в очереди нет других элементов на местах, кратных 4).

Для n = 3 метод вернет false , потому что элемент на месте 3 – это не "магический элемент".

/*продолжение на странице 6*/

5. Дан класс **Patient** – пациент в приемном отделении; в этом классе два атрибута [תכונות]:

- id – номер удостоверения личности пациента, целочисленного типа
- priority – степень срочности оказания помощи пациенту. Степень срочности представлена целым числом от 1 до 10 . Чем больше это число, тем выше степень срочности.

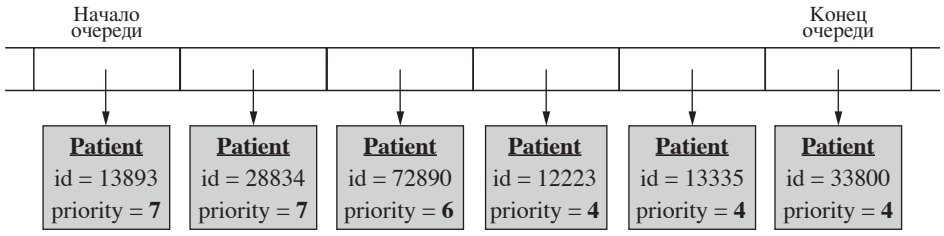
Считайте, что для атрибутов этого класса определены методы get/Get и set/Set .

Порядок оказания помощи пациентам в приемном отделении организован следующим образом: чем выше степень срочности оказания помощи, тем раньше пациент получает помощь. Когда есть более одного пациента с одинаковой степенью срочности, то пациент, который прибыл в приемное отделение раньше, получает помощь раньше.

Чтобы соблюдать порядок оказания помощи, создан класс **PriorQueue** – очередь приоритетов; в этом классе один атрибут:

- q – указатель [הפניה] на очередь, типа Patient .

Пример очереди q :



Считайте, что пациенты с одинаковой степенью срочности представлены в примере в порядке их прибытия в приемное отделение.

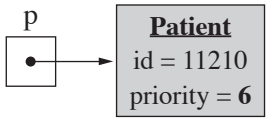
(א) Реализуйте следующий метод, принадлежащий классу PriorQueue :

Java – public void priorityInsert (Patient p)

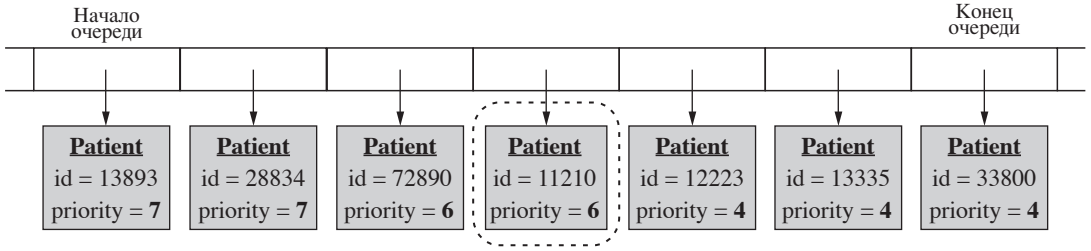
C# – public void PriorityInsert (Patient p)

Метод получает нового пациента p и добавляет его в очередь q в соответствии с правилами приемного отделения, написанными выше.

Например: для очереди, представленной выше, и следующего объекта:



После добавления очередь будет выглядеть так:



/продолжение на странице 7/

(ב) Иногда степень срочности для определенного пациента изменяется во время его пребывания в приемном отделении.

Реализуйте следующий метод, принадлежащий классу PriorQueue :

Java – public void update (int id, int pri)

C# – public void Update (int id, int pri)

Метод получает номер удостоверения личности пациента, находящегося в очереди, id и число pri , представляющее обновленную степень срочности оказания помощи пациенту. Этот метод обновит атрибут priority пациента и переместит его в очереди для оказания помощи на соответствующее ему место (согласно обновленной степени срочности – pri). Если в очереди уже есть другие пациенты с той же степенью срочности pri , текущий пациент id будет помещен после них, как если бы он прибыл в приемное отделение после них.

/продолжение на странице 8/

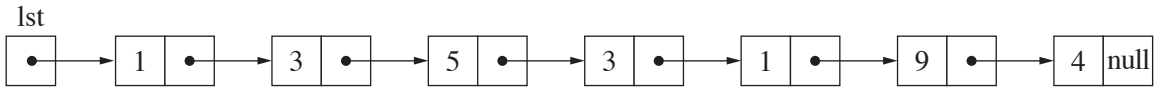
6. Обратите внимание**:** этот вопрос имеет две версии: на языке Java на странице 8 и на языке C# на странице 9.

Для пишущих на языке Java

(א) Дан метод what :

```
public static Node<Integer> what (Node<Integer>lst, int x)
{
    if (lst == null)
        return null;
    Node<Integer> temp = what (lst.getNext(),x);
    if (lst.getValue() == x)
        return temp;
    lst.setNext (temp);
    return lst;
}
```

Дана цепочка звеньев lst , целочисленного типа:



(1) Напишите трассировку [עקבו] метода what (lst, 1) и приведите цепочку, которую метод возвращает.

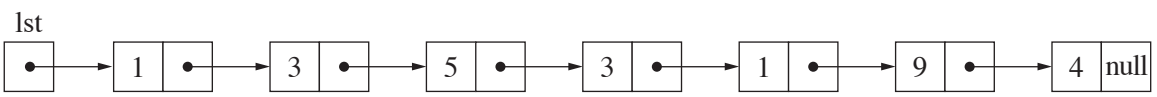
(2) Что делает метод what ? Объясните свой ответ.

(3) Какова сложность [סיבוכיות] метода what ? Обоснуйте свой ответ.

(ב) Дан метод guess :

```
public static void guess (Node<Integer>lst)
{
    if (lst != null)  {
        Node<Integer> temp = what (lst.getNext(), lst.getValue());
        lst.setNext (temp);
        guess (lst.getNext());
    }
}
```

Дана цепочка звеньев lst , целочисленного типа:



(1) Напишите трассировку [עקבו] метода guess (lst) и приведите цепочку lst по окончании этого метода.

В этом пункте нет необходимости в трассировке метода what .

(2) Что делает метод guess ? Объясните свой ответ.

(3) Какова сложность [סיבוכיות] метода guess ? Обоснуйте свой ответ.

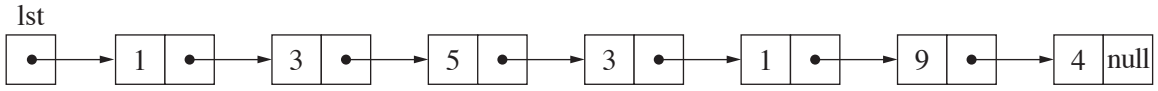
/продолжение на странице 9/

Для пишущих на языке C#

(א) Дан метод What :

```
public static Node<int> What (Node<int>lst, int x)
{
    if (lst == null)
        return null;
    Node<int> temp = What (lst.GetNext(),x);
    if (lst.GetValue() == x)
        return temp;
    lst.SetNext (temp);
    return lst;
}
```

Дана цепочка звеньев lst , целочисленного типа:



(1) Напишите трассировку [עקבו] метода What (lst, 1) и приведите цепочку, которую метод возвращает.

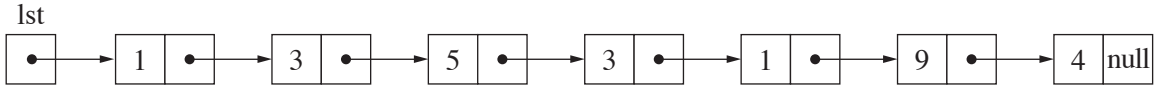
(2) Что делает метод What ? Объясните свой ответ.

(3) Какова сложность [סיבוכיו] метода What ? Обоснуйте свой ответ.

(ב) Дан метод Guess :

```
public static void Guess (Node<int>lst)
{
    if (lst != null)  {
        Node<int> temp = What (lst.GetNext(), lst.GetValue());
        lst.SetNext (temp);
        Guess (lst.GetNext());
    }
}
```

Дана цепочка звеньев lst , целочисленного типа:



(1) Напишите трассировку [עקבו] метода Guess (lst) и приведите цепочку lst по окончании этого метода.

В этом пункте нет необходимости в трассировке метода What .

(2) Что делает метод Guess ? Объясните свой ответ.

(3) Какова сложность [סיבוכיות] метода Guess ? Обоснуйте свой ответ.

/продолжение на странице 10/

7. Дан класс **BusStation** – автобусная остановка; в этом классе три атрибута:

- num – номер остановки, целочисленного типа.
- arr – массив целочисленного типа длиной 10 , содержащий номера маршрутов автобусов, которые останавливаются на этой остановке. На остановке останавливается до 10 автобусных маршрутов. Ими заполнен массив, с начала массива и непрерывно.
- amount – количество автобусных маршрутов, которые фактически останавливаются на остановке, целочисленного типа. На остановке останавливается по меньшей мере один автобусный маршрут.

Считайте, что для атрибутов этого класса определены методы get/Get и set/Set .

(א) Реализуйте приведенный ниже метод, принадлежащий интефейсу класса BusStation :

Java – public boolean isStopping (int n)
C# – public bool IsStopping (int n)

Метод принимает номер автобусного маршрута n , целочисленного типа. Метод возвращает true , если автобусный маршрут останавливается на остановке, иначе метод возвращает false.

(ב) Дан массив arr типа BusStation , и в нем все автобусные остановки одного города. Известно, что несколько автобусных маршрутов останавливаются на каждой остановке этого города, а остальные автобусные маршруты останавливаются только на части остановок этого города.
Напишите внешний метод [פעולה חיצונית] с названием allStations на языке Java или AllStations на языке C# , который принимает массив arr .
Этот метод возвращает цепочку звеньев целочисленного типа, в каждом звене которой хранится один из номеров маршрутов, которые останавливаются на **всех** остановках этого города (каждый такой маршрут сохраняется в цепочке только один раз).
Примечание: порядок маршрутов в цепочке не имеет значения.

/продолжение на странице 11/

РАЗДЕЛ ТРЕТИЙ

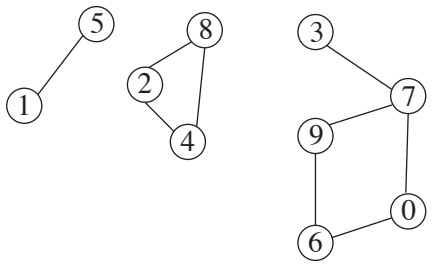
В этом разделе вопросы из трех курсов:
Алгоритмы [אלגוריתמים], страницы 11–12,
Вычислительные модели [מודלים חישוביים], страницы 13–14,
Объектно-ориентированное программирование [תכנות מונחה עצמים] на языке Java , страницы 15–18;
Объектно-ориентированное программирование на языке C# , страницы 19–22.
Выберите вопросы только одного курса.

Алгоритмы [אלגוריתמים]

8. Дан граф, несвязный [לא קשיר] и неориентированный [לא מכוון] $G(V, E)$, в котором n вершин, от v_0 до v_{n-1} .

(א) Напишите алгоритм, который находит и возвращает все вершины, у которых есть маршрут между вершиной в графе v_j и ними.
Примечание: нужно написать эффективный алгоритм, который не проходит по всем возможным маршрутам.
Пример: для графа, приведенного ниже, и вершины 3 алгоритм вернет вершины 0 , 6 , 7 , 9 .

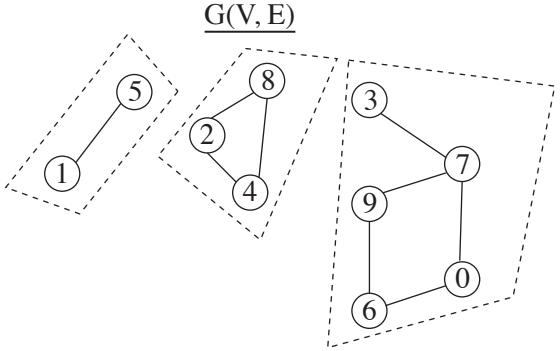
G(V, E)



Объяснение: существует маршрут между вершиной 3 и вершинами 0 , 6 , 7 , 9 .

(ב) "Компонент связности" [רכיב קשירות"] в неориентриованном графе $G(V, E)$ – это группа вершин, в которой между каждыми двумя вершинами есть маршрут, и нет никакого ребра, выходящего из вершины, находящейся в этой группе, к вершине вне группы. Вершина, для которой нет ребра между ней и любой другой вершиной, будет находиться в своей собственной группе.
Пример: для графа, приведенного выше, три компонента связности обозначены пунктиром.

G(V, E)



Напишите алгоритм, который находит в графе $G(V, E)$ и возвращает самый маленький компонент связности (иными словами, группу с минимальным числом вершин).
Так, для примера, приведенного выше, алгоритм вернет вершины 1 , 5 .
Считайте, что есть только один компонент связности, который является самым маленьким.
Примечание: нужно написать эффективный алгоритм, который не проходит по всем возможным маршрутам.

/продолжение на странице 12/

9. Ниже приведены шесть утверждений (א)–(ו). Выберите пять из них и укажите относительно каждого выбранного утверждения, верно ли оно. Если утверждение верно, обоснуйте почему. Если утверждение неверно, приведите опровергающий пример.

(א) Любое дерево, полученное при применении DFS на неориентированном [לא מכוון] графе G, может получиться также при применении BFS на том же графе.

(ב) Любое дерево, полученное при применении DFS на неориентированном полном [לא מכוון מלא] графе G – это дерево, в котором у каждой вершины [צומת] есть только один сын.

(ג) Дан ориентированный граф G и вершина v. Если можно добраться от вершины v до каждой из вершин графа, а также можно добраться из каждой вершины до вершины v, то граф G – это обязательно сильно связный [קשיר היטב] граф.

(ד) Если в неориентированном графе G самый короткий маршрут от вершины s_j до вершины s_n – это $S[s_j \ldots \mathbf{s_m} \ldots \mathbf{s_k} \ldots s_n]$, то внутренний маршрут от s_m до s_k – обязательно самый короткий маршрут от вершины s_m до вершины s_k.

(ה) Граф, в котором есть цикл, не может быть двудольным графом (биграф) [גרף דו־צדדי].

(ו) У любого взвешенного [ממושקל] неориентированного графа есть единственное минимальное покрывающее дерево [עץ פורש מינימלי].

/продолжение на странице 13/

Вычислительные модели [מודלים חישוביים]

10. В этом вопросе два пункта (א)–(ב). Между ними нет связи. Ответьте на вопросы обоих пунктов.

(א) Ниже приведены два утверждения (1)–(2) относительно языков L_2 , L_1 над алфавитом $\{a, b\}$.

Для каждого утверждения укажите, верно ли оно. Если утверждение верно, обоснуйте почему. Если оно не верно, приведите опровергающий пример. Между данными утверждениями нет связи.

(1) Если L_2 , L_1 – нерегулярные языки, то $L_1 \cap L_2$ – непременно нерегулярный язык.

(2) $L_1^n \cdot L_2^n$ всегда равен $(L_1 \cdot L_2)^n$.

(ב) Дан язык L над алфавитом $\{a, b, c\}$:

$L = \{w \mid$ последовательности bc , ab не появляются в w , $\#_a(w) + \#_c(w) =$ четное число $\}$

$\#_a(w)$ указывает число появлений a в слове w .

$\#_c(w)$ указывает число появлений c в слове w .

Пример слова в языке L – это cba , потому что сумма появлений буквы a (1) и буквы c (1) – четное число (2) и последовательности ab и bc не появляются в этом слове.

(1) Ниже приведено пять слов. Относительно каждого из них укажите, принадлежит ли это слово языку L . Обоснуйте свой ответ.

$cbbba$, baa , ε , aab , $aaccc$.

(2) Постройте неполный детерминированный конечный автомат, принимающий язык L .

/продолжение на странице 14/

11. В этом вопросе два пункта (א)–(ב). Между ними нет связи. Ответьте на вопросы обоих пунктов.

(א) Ниже приведены шесть языков над алфавитом $\{0,1\}$:

Σ^* обозначает язык всех слов над алфавитом $\{0,1\}$.

$L_1 = \emptyset$ $\quad L_4 = \{0110\}$

$L_2 = \Sigma^*$ $\quad L_5 = \{\varepsilon, 110, 00, 001\}$

$L_3 = \{\varepsilon\}$ $\quad L_6 = \{1, 0110, 110, 01\}$

Напишите язык, который получается в результате каждого из пяти следующих действий:

(1) $L_5 \cap L_6$

(2) L_2^R

(3) $L_1 \cdot L_6$

(4) $L_3 \cdot L_4$

(5) $L_4 \cdot L_5$

(ב) Дан язык L над алфавитом $\{a, b, c\}$:

$$L = \left\{ (ab)^k \, c^m \, b^{m+3k} \mid k, m >= 0 \right\}$$

Постройте детерминированный стек-автомат, который принимает язык L.

/продолжение на странице 15/

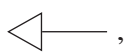
Объектно-ориентированное программирование [תכנות מונחה עצמים] на языке Java

12. В компании по аренде транспортных средств "Удачная поездка" разработана компьютеризированная система, в которой есть следующие классы:

Contract – Договор, **Vehicle** – Транспортное средство, **Car** – Автомобиль, **Truck** – Грузовик, **Motorcycle** – Мотоцикл.

Ниже перечислены атрибуты этих классов.

- У класса Договор (Contract) три атрибута: name – имя клиента (строка), days – число дней аренды (целое число), kilo – число километров, которые проехал клиент (целое число).
- У класса Транспортное средство (Vehicle) два атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract).
- У класса Автомобиль (Car) три атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract), число посадочных мест – seats (целое число).
- У класса Грузовик (Truck) три атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract), максимальный вес груза – max (целое число).
- У класса Мотоцикл (Motorcycle) три атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract), мотоцикл-внедорожник – offRoad (булева типа. Если мотоцикл – внедорожник, значение атрибута – правда; если нет, то значение атрибута – ложь).

(א) **(1)** Начертите схему иерархии, описывающую связь между классами этой компьютеризированной системы.
Обозначьте наследование посредством стрелки  ,
а включение – посредством символа  .

(2) Напишите названия классов и их атрибуты. Считайте, что методы get и set определены для всех атрибутов классов и нет необходимости их реализовывать [לממש].

(ב) Объявлен конструктор [פעולה בונה] класса Contract:
public Contract (String name, int days, int kilo)
Нет необходимости реализовывать этот конструктор.

(1) Ниже объявлен конструктор класса Vehicle. Конструктор получает имя клиента, число дней аренды, число километров, которые проехал клиент, и идентификатор транспортного средства.
public Vehicle (String name, int days, int kilo, String id)
Реализуйте этот конструктор.

(2) Ниже объявлен конструктор класса Car. Конструктор получает имя клиента, число дней аренды, число километров, которые проехал клиент, идентификатор транспортного средства и число посадочных мест.
public Car (String name, int days, int kilo, String id, int seats)
Реализуйте этот конструктор.

/продолжение на странице 16/

Базовый тариф аренды транспортного средства – Vehicle, который берет эта компания – 60 шекелей за день аренды и 2 шекеля за каждый километр поездки.

Цена аренды автомобиля (Car) соответствует базовому тарифу.

Цена аренды грузовика (Truck) соответствует базовому тарифу, и дополнительно к нему платится одноразовая сумма в размере 500 шекелей.

Цена аренды мотоцикла (Motorcycle) равна половине базового тарифа.

(ג) Метод payment возвращает действительное число, равное сумме, которую клиент должен заплатить за аренду каждого из типов траспортных средств, которые сдает эта компания (в зависимости от объекта, который вызвал этот метод).

(1) Напишите метод payment в классе Vehicle (как сказано выше, сумма платежа – в соответствии с базовым тарифом).

(2) Добавьте метод payment к другому классу/классам, чтобы выполнить требование (только в классах, в которых есть такая необходимость) согласно принципам объекто-ориентированного программирования.

Примечание: в этом пункте **запрещено** использовать метод instanceOf и методы класса Object и запрещено изменять атрибуты классов.

Решение, в котором они использовались, не будет засчитано.

/продолжение на странице 17/

13. Ниже объявлены классы AA , BB и их атрибуты.

Конструкторы классов обозначены *** (возможно более одного конструктора в классе).

```
public class AA
{
    private int x;
    ***
}
public class BB extends AA
{
    private AA f;
    ***
}
```

Примечание: **обратите внимание – у обоих классов нет методов** get и set .

Ниже приведен класс Test , включающий главный метод:

```
public class Test{
   public static void main(String[] args) {
        AA a1 = new AA ();
        AA a2  = new AA (8);
        AA a3 = new AA (3);
        AA a4 = new AA (a1);
        AA a5 = new AA (a2);
        BB b1 = new BB ();
        BB b2 = new BB (6, a4);
        BB b3 = new BB (5, a2);
        BB b4 = new BB (a4);
        BB b5 = new BB (a2);
   }
}
```

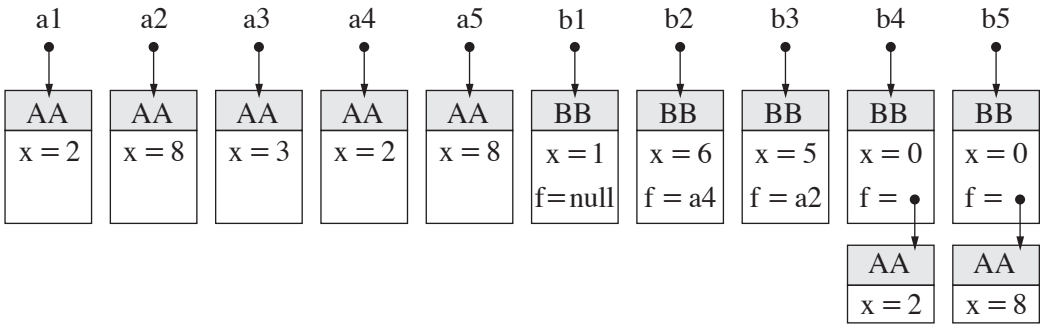
/продолжение на странице 18/

Ниже приведена схема объектов, которые были созданы вследствие запуска фрагмента кода.

Напишите в классах AA и BB конструкторы, требуемые для получения объектов этой схемы.

Для каждого созданного объекта укажите подходящий ему конструктор.

Примечание: **запрещается** добавлять методы, которые не являются конструкторами в классах AA и BB . Решение, которое добавляет методы, не являющиеся конструкторами, не будет засчитано.



/продолжение на странице 19/

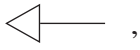
Объектно-ориентированное программирование [תכנות מונחה עצמים] на языке C#

14. В компании по аренде транспортных средств "Удачная поездка" разработана компьютеризированная система, в которой есть следующие классы:

Contract – Договор, **Vehicle** – Транспортное средство, **Car** – Автомобиль, **Truck** – Грузовик, **Motorcycle** – Мотоцикл.

Ниже перечислены атрибуты этих классов.

- У класса Договор (Contract) три атрибута: name – имя клиента (строка), days – число дней аренды (целое число), kilo – число километров, которые проехал клиент (целое число).
- У класса Транспортное средство (Vehicle) два атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract).
- У класса Автомобиль (Car) три атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract), число посадочных мест – seats (целое число).
- У класса Грузовик (Truck) три атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract), максимальный вес груза – max (целое число).
- У класса Мотоцикл (Motorcycle) три атрибута: идентификатор транспортного средства – id (строка), договор – contract (Contract), мотоцикл-внедорожник – offRoad (булева типа. Если мотоцикл – внедорожник, значение атрибута – правда; если нет, то значение атрибута – ложь).

(א) **(1)** Начертите схему иерархии, описывающую связь между классами этой компьютеризированной системы.
Обозначьте наследование посредством стрелки  ,
а включение – посредством символа  .

(2) Напишите названия классов и их атрибуты. Считайте, что методы Get и Set определены для всех атрибутов классов и нет необходимости их реализовывать [לממש].

(ב) Определен конструктор [פעולה בונה] класса Contract:
public Contract (string name, int days, int kilo)
Нет необходимости реализовывать этот конструктор.

(1) Ниже объявлен конструктор класса Vehicle. Конструктор получает имя клиента, число дней аренды, число километров, которые проехал клиент, и идентификатор транспортного средства.
public Vehicle (string name, int days, int kilo, string id)
Реализуйте этот конструктор.

(2) Ниже объявлен конструктор класса Car. Конструктор получает имя клиента, число дней аренды, число километров, которые проехал клиент, идентификатор транспортного средства и число посадочных мест.
public Car (string name, int days, int kilo, string id, int seats)
Реализуйте этот конструктор.

/продолжение на странице 20/

Базовый тариф аренды транспортного средства – Vehicle, который берет эта компания – 60 шекелей за день аренды и 2 шекеля за каждый километр поездки.

Цена аренды автомобиля (Car) соответствует базовому тарифу.

Цена аренды грузовика (Truck) соответствует базовому тарифу, и дополнительно к нему платится одноразовая сумма в размере 500 шекелей.

Цена аренды мотоцикла (Motorcycle) равна половине базового тарифа.

(ג) Метод Payment возвращает действительное число, равное сумме, которую клиент должен заплатить за аренду каждого из типов траспортных средств, которые сдает эта компания (в зависимости от объекта, который вызвал этот метод).

(1) Напишите метод Payment в классе Vehicle (как сказано выше, сумма платежа – в соответствии с базовым тарифом).

(2) Добавьте метод Payment к другому классу/классам, чтобы выполнить требование (только в классах, в которых есть такая необходимость) согласно принципам объекто-ориентированного программирования.

Примечание: в этом пункте **запрещено** использовать методы is и as и методы класса Object и запрещено изменять атрибуты классов.

Решение, в котором они использовались, не будет засчитано.

/продолжение на странице 21/

15. Ниже объявлены классы AA , BB и их атрибуты.

Конструкторы классов обозначены *** (возможно более одного конструктора в классе).

```
public class AA
{
    private int x;
    ***
}
public class BB : AA
{
    private AA f;
    ***
}
```

Примечание: **обратите внимание – у обоих классов нет методов** Get и Set .

Ниже приведен класс Test , включающий главный метод:

```
public class Test{
    public static void Main(string[] args) {
        AA a1 = new AA ();
        AA a2  = new AA (8);
        AA a3 = new AA (3);
        AA a4 = new AA (a1);
        AA a5 = new AA (a2);
        BB b1 = new BB ();
        BB b2 = new BB (6, a4);
        BB b3 = new BB (5, a2);
        BB b4 = new BB (a4);
        BB b5 = new BB (a2);
    }
}
```

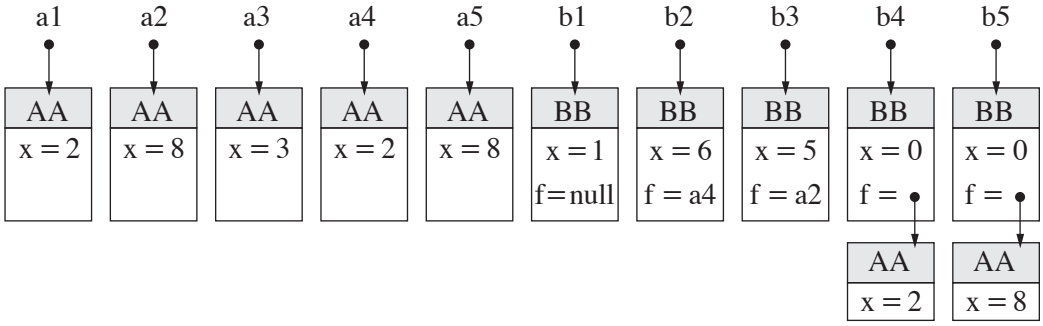
/продолжение на странице 22/

Ниже приведена схема объектов, которые были созданы вследствие запуска фрагмента кода.

Напишите в классах AA и BB конструкторы, требуемые для получения объектов этой схемы.

Для каждого созданного объекта укажите подходящий ему конструктор.

Примечание: **запрещается** добавлять методы, которые не являются конструкторами в классах AA и BB . Решение, которое добавляет методы, не являющиеся конструкторами, не будет засчитано.



Желаем успеха!

בהצלחה!