

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

מדעי המחשב

Указания

הוראות

- א. Продолжительность экзамена: 3 часа 30 минут.
- ב. Строение вопросника и ключ к оценке:
в этом вопроснике два раздела.
Нужно выбрать в общей сложности четыре вопроса.
За каждый вопрос – 25 баллов; всего 100 баллов.
Обратите внимание: если вы решите отвечать на вопросы второго раздела, выберете вопросы только из одного курса.
- в. Разрешенный вспомогательный материал:
любой вспомогательный материал, за исключением калькулятора с возможностью программирования.
- г. Особые указания:
- (1) Если вы решите отвечать на вопросы **только** первого раздела, **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.

- א. משך הבחינה: שלוש שעות וחצי.
ב. מבנה השאלון ומפתח ההערכה:
בשאלון זה שני פרקים.
יש לבחור בארבע שאלות סך הכול.
לכל שאלה – 25 נקודות; סך הכול – 100 נקודות.
שימו לב: אם תבחרו לענות על שאלות מן הפרק השני, בחרו בשאלות מתוך מסלול אחד בלבד.
- ג. חומר עזר מותר בשימוש:
כל חומר עזר, חוץ ממחשבון שיש בו אפשרות תכנות.
- ד. הוראות מיוחדות:
- (1) אם תבחרו לענות על שאלות **רק** מן הפרק הראשון **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 **כתבו 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!

בהצלחה!

В о п р о с ы

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

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

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);
```

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.

РАЗДЕЛ ПЕРВЫЙ

1. "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".

- (*) 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 ($11 + 9 \neq 6$)

Обратите внимание: продолжение вопроса на следующей странице.

- (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".

2. 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.

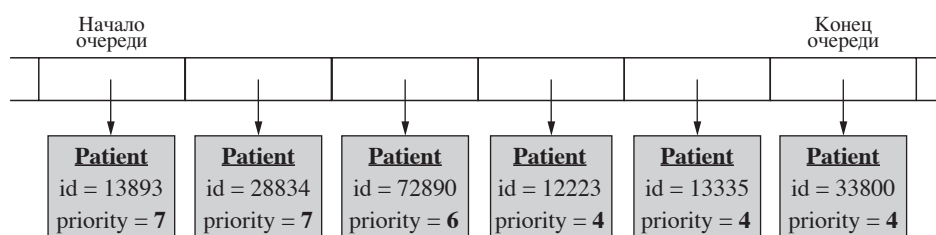
Считайте, что для атрибутов этого класса определены методы get/Get и set/Set.

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.

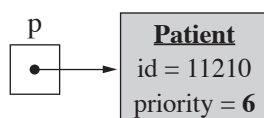
(*) Реализуйте следующий метод, принадлежащий классу PriorQueue :

Java – public void priorityInsert (Patient p)

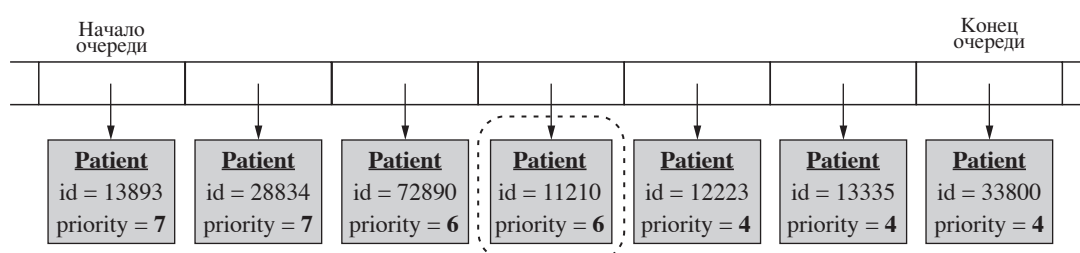
C# – public void PriorityInsert (Patient p)

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:



- (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.

3. Обратите внимание: этот вопрос имеет две версии: на языке Java на странице 6 и на языке C# на странице 7.

Для пишущих на языке Java

(⌘) 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 [עקבון] 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.
- (⌘) 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 [עקבון] 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.

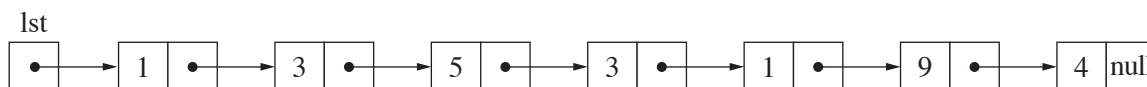
/продолжение на странице 7/

Для пишущих на языке C#

(8) 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:

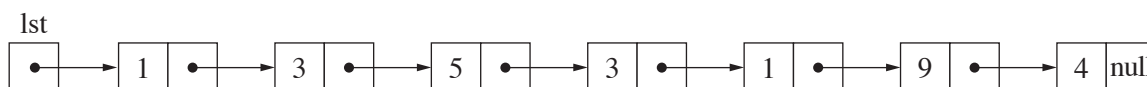


- (1) Напишите трассировку [טרקור] метода What (lst, 1) и приведите цепочку, которую метод возвращает.
- (2) Что делает метод What ? Объясните свой ответ.
- (3) Какова сложность [סיבוכיות] метода What ? Обоснуйте свой ответ.

(9) 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 [טרקור] 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.

4. 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 .

(*) Реализуйте приведенный ниже метод, принадлежащий интерфейсу 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.

(*) 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 [פועל חיצוני] 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.

5. "Противоположная подпоследовательность" – ряд чисел, который начинается с любого числа и заканчивается противоположным по знаку и равным ему по модулю числом (иными словами, начинается положительным числом и заканчивается отрицательным числом, соответствующим ему, или начинается отрицательным числом и заканчивается положительным числом, соответствующим ему). Этот ряд находится внутри последовательности чисел.

Например: в последовательности чисел 17, 3, 1, -4, 5, 6, 4, -3, 7, 23, -99 есть две противоположные подпоследовательности:

i. 3, 1, -4, 5, 6, 4, -3

ii. -4, 5, 6, 4

Дана цепочка звеньев `lst` целочисленного типа, содержащая положительные и отрицательные числа, которые не равны 0 и все отличны друг от друга (иными словами, нет двух одинаковых чисел в цепочке).

- (*) Напишите метод с названием `width` на языке Java и `Width` на языке C#, который принимает цепочку `lst` и целое число `num` (положительное или отрицательное), появляющееся в цепочке.

Метод возвращает длину "противоположной подпоследовательности", которую число `num` **начинает или заканчивает** (длина включает числа по краям).

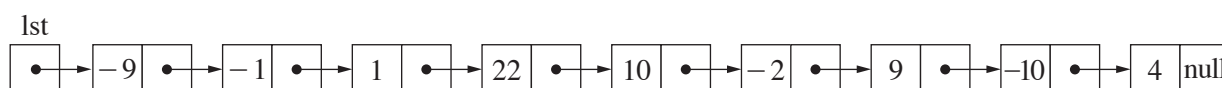
Если число, противоположное `num`, не появляется в цепочке, метод возвращает `-1`.

Примечания: – необходимо сохранить цепочку `lst`.

– В этом пункте **запрещено** пользоваться массивом.

Решение, в котором массив использовался, не будет засчитано.

Примеры: для цепочки `lst`, приведенной ниже:



Для `num = 9` метод возвращает 7 (подпоследовательность: -9, -1, 1, 22, 10, -2, 9).

Для `num = -1` метод возвращает 2 (подпоследовательность: -1, 1).

Для `num = 22` метод возвращает `-1` (потому что противоположное ему число, `-22`, не появляется в цепочке).

- (*) Напишите метод `longest` на языке Java или `Longest` на языке C#, который принимает цепочку `lst`. Метод возвращает длину самой большой "противоположной подпоследовательности". Если в цепочке нет ни одной "противоположной подпоследовательности", метод возвращает `-1`.

Можно воспользоваться методом, который вы написали в пункте (*).

Примечания: – в этом методе не обязательно сохранять цепочку `lst`.

– В этом пункте **запрещено** пользоваться массивом.

Решение, в котором массив использовался, не будет засчитано.

Пример: для цепочки `lst`, приведенной в примере выше, метод вернет 7.

Объяснение: "противоположная подпоследовательность", которая начинается с числа `-9` и заканчивается числом `9`, содержит семь чисел и является самой большой.

/продолжение на странице 10/

РАЗДЕЛ ВТОРОЙ

В этом разделе вопросы из трех курсов:

Алгоритмы [אלגוריתמים], страницы 10–11,

Вычислительные модели [מודלים חישוביים], страницы 12–14,

Объектно-ориентированное программирование [תכנות מונחה עצמים] на языке Java, страницы 15–20;

Объектно-ориентированное программирование на языке C#, страницы 21–26.

Выберите вопросы только одного курса.

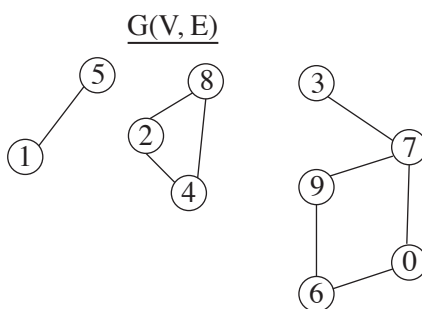
Алгоритмы [אלגוריתמים]

6. 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.

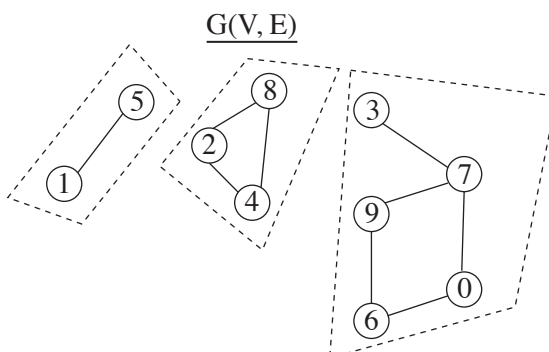
Пример: для графа, приведенного ниже, и вершины 3 алгоритм вернет вершины 0, 6, 7, 9.



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

(ב) "Компонент связности" [רכיב קשירות] в неориентированном графе $G(V, E)$ – это группа вершин, в которой между любыми 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.



Напишите алгоритм, который находит в графе $G(V, E)$ и возвращает самый маленький компонент связности (иными словами, группу с минимальным числом вершин).

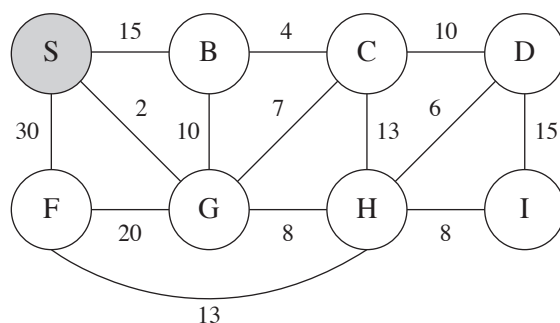
Так, для примера, приведенного выше, алгоритм вернет вершины 1, 5.

Считайте, что есть только один компонент связности, который является самым маленьким.

Примечание: нужно написать эффективный алгоритм, который не проходит по всем возможным маршрутам.

7. 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 [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 $S[s_j \dots s_m \dots s_k \dots s_n]$, 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 מורש מינימלי].

8. Ниже приведен взвешенный граф G .



- (א) (1) Напишите алгоритм, который находит в любом **взвешенном** графе (не отрицательном), в котором n вершин от v_0 до v_{n-1} и v_j – вершина графа, самые короткие (легкие) маршруты от вершины графа v_j до остальных вершин графа.
- (2) Какова временная сложность [סיבוכיות זמן הריצה] этого алгоритма? Обоснуйте свой ответ.
- (ב) (1) Для данного графа G найдите с помощью написанного вами алгоритма самый короткий маршрут от вершины S до каждой из вершин и начертите таблицу трассировки [טבלת מעקב], как объяснено ниже:
в каждую итерацию трассировка будет включать группу постоянных вершин (которые мы уже посетили), P , и группу временных вершин (которые мы еще не посетили), T . Кроме того, для каждой вершины будет указана длина маршрута до нее и идентифицирована вершина, предшествующая ей (ее "родитель").
- (2) Для данного графа G начертите дерево кратчайших маршрутов (из вершины S).

Вычислительные модели [מודלים חישוביים]

9. 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 $\{a, b, c\}$:

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

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

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

Пример слова в языке L – это sba , 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 .

10. 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) $L_3 \cdot L_4$

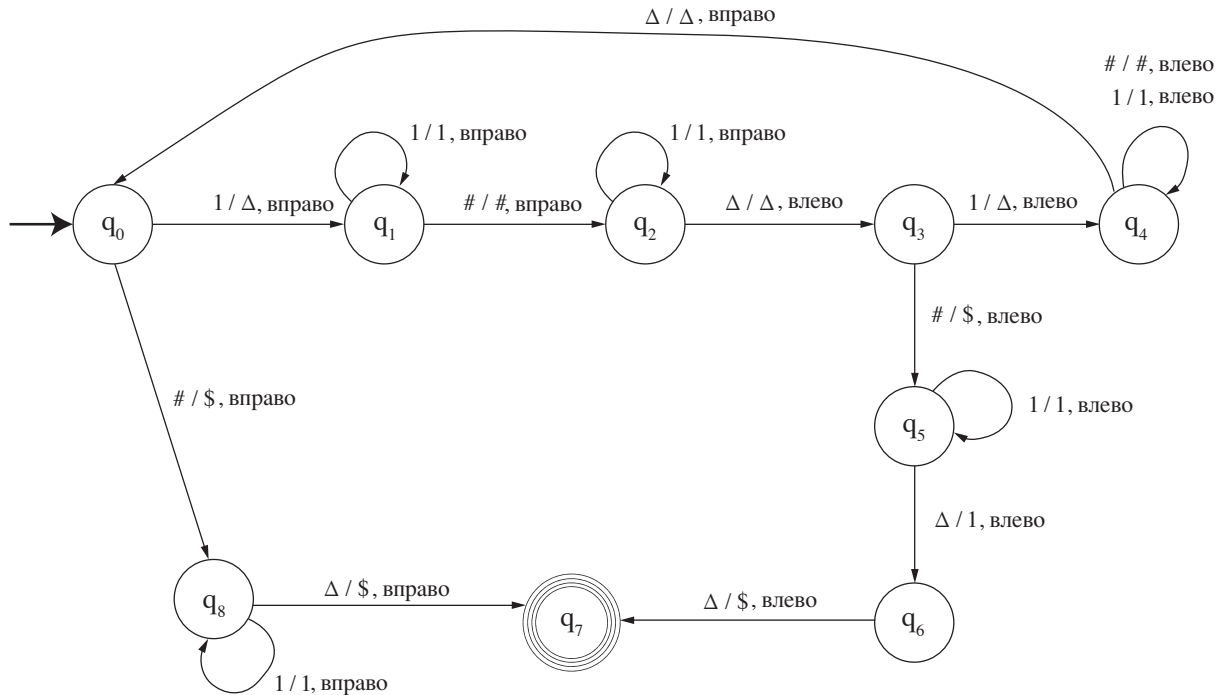
(5) $L_4 \cdot L_5$

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

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

Постройте детерминированный стек-автомат, который принимает язык L .

11. Ниже приведена машина Тьюринга:



Машина получает в начале ленты два унарных числа [מספרים אונריים], между которыми есть знак #.

(א) Покажите трассировку [תדבור] машины для следующей ленты:

⊢	1	#	1	1	1	Δ	Δ	...
---	---	---	---	---	---	---	---	-----

Напишите, как будет выглядеть лента после завершения работы машины.

(ב) Напишите, как будет выглядеть лента после завершения работы машины:

⊢	1	1	#	1	Δ	Δ	...
---	---	---	---	---	---	---	-----

Примечание: в этом пункте нет необходимости показывать трассировку [מעקב].

(ג) Что выполняет машина для двух унарных чисел, которые больше нуля?

(ד) Ниже приведены три случая, 1–3. Для каждого случая укажите, правильно или неправильно действует машина.

Если машина действует правильно, покажите на примере ленту ввода [סרט הקלט] и ленту вывода [סרט הפלט]. Если машина действует неправильно, объясните почему.

1. Первое число (слева) равно нулю, а второе число больше нуля.
2. Второе число равно нулю, а первое число больше нуля.
3. Оба числа равны нулю.

Объектно-ориентированное программирование [תכנות מונחם] 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).

(*) (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].

(*) 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.

Обратите внимание: продолжение вопроса на следующей странице.

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.

- (a) Метод `payment` возвращает действительное число, равное сумме, которую клиент должен заплатить за аренду каждого из типов транспортных 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.

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);
    }
}
```

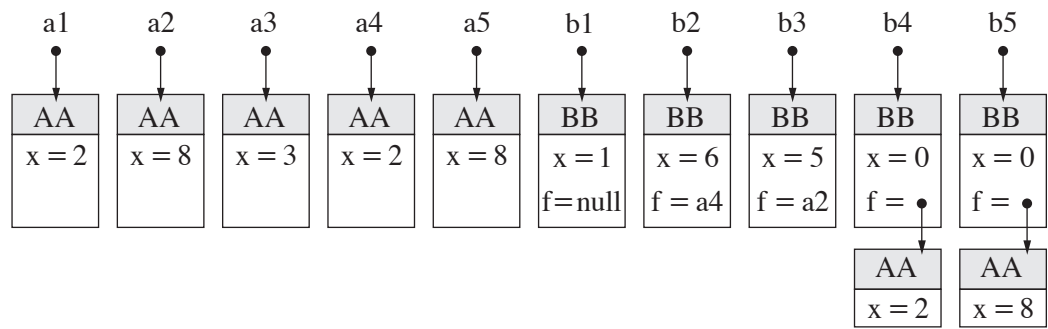
Обратите внимание: продолжение вопроса на следующей странице.

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.



14. Даны классы AA , BB :

```
public class AA {  
    private int x;  
  
    public AA() { this.x = 4; }  
    public AA(int x) { this.x = x; }  
    public int getX() { return this.x; }  
    public void foo() { this.x = 3; }  
    public void goo() { this.x = this.x + 3; }  
    public void bar() { goo(); }  
    public String toString() { return "x = " + this.x; }  
}
```

```
public class BB extends AA {  
    private int y;  
  
    public BB() {  
        super();  
        this.y = 2;  
    }  
    public BB (int a) {  
        super (a);  
        this.y = getX() * 2;  
    }  
    public BB (int a, int b) {  
        super (a);  
        this.y = b;  
    }  
    public int getY() { return this.y; }  
    public void foo() { this.y = 5; }  
    public void goo() { this.y = getX() - 1; }  
    public void bar (int a) {  
        this.y = a + getX() + this.y;  
    }  
    public String toString() {  
        return super.toString() + " y = " + this.y;  
    }  
}
```

Обратите внимание: продолжение вопроса на следующей странице.

(א) Ниже приведен фрагмент кода:

```
AA[] items = new AA[6];  
items[0] = new AA();  
items[1] = new BB();  
items[2] = new AA(2);  
items[3] = new BB(2);  
items[4] = new BB(1, 22);  
items[5] = items[4];  
for (int i = 0; i < items.length; i++)  
    System.out.println(items[i]);
```

Нарисуйте созданные объекты и напишите, что код выводит на экран.

(ב) Ниже приведено продолжение этого фрагмента кода:

```
items[0].foo();  
items[1].goo();  
((AA) items[3]).goo();  
items[4].bar();  
BB temp = new BB(1);  
temp.bar();  
System.out.println("Temp: " + temp);  
temp.bar(9);  
items[2] = temp;  
for (int i = 0; i < items.length; i++)  
    System.out.println(items[i]);
```

Напишите, что код выводит на экран.

Объектно-ориентированное программирование [ע"מזחזח ע"מזחזח] на языке C#

15. 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).

(*) (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 [לממז].

(*) 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.

Обратите внимание: продолжение вопроса на следующей странице.

/продолжение на странице 22/

Компьютеры, лето 2024 года, № 899271

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.

(a) Метод Payment возвращает действительное число, равное сумме, которую клиент должен заплатить за аренду каждого из типов транспортных 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.

16. 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);
    }
}
```

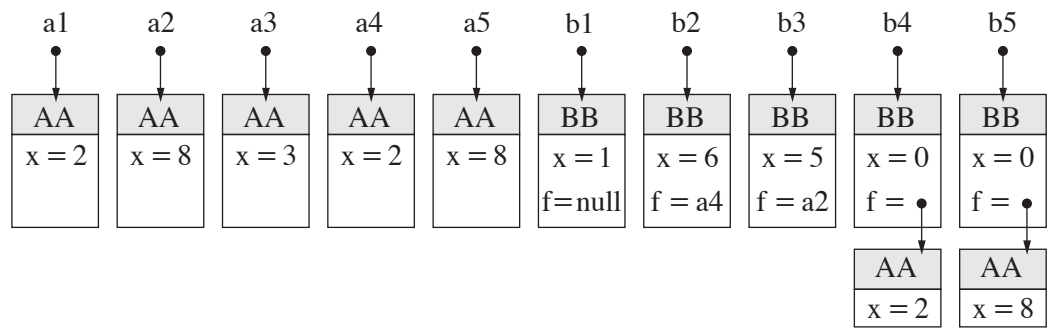
Обратите внимание: продолжение вопроса на следующей странице.

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.



17. Даны классы AA , BB :

```
public class AA {  
    private int x;  
  
    public AA() { this.x = 4; }  
    public AA(int x) { this.x = x; }  
    public int GetX() { return this.x; }  
    public virtual void Foo() { this.x = 3; }  
    public virtual void Goo() { this.x = x + 3; }  
    public void Bar() { Goo(); }  
    public override string ToString() { return "x = " + this.x; }  
}
```

```
public class BB : AA {  
    private int y;  
  
    public BB() : base() {  
        this.y = 2;  
    }  
    public BB (int a) : base (a) {  
        this.y = GetX() * 2;  
    }  
    public BB(int a, int b) : base(a) {  
        this.y = b;  
    }  
    public int GetY() { return this.y; }  
    public override void Foo() { this.y = 5; }  
    public override void Goo() { this.y = GetX() - 1; }  
    public void Bar (int a) {  
        this.y = a + GetX() + this.y;  
    }  
    public override string ToString() {  
        return base.ToString() + " y = " + this.y;  
    }  
}
```

Обратите внимание: продолжение вопроса на следующей странице.

(8) Ниже приведен фрагмент кода:

```
AA[] items = new AA[6];
items[0] = new AA();
items[1] = new BB();
items[2] = new AA(2);
items[3] = new BB(2);
items[4] = new BB(1, 22);
items[5] = items[4];
for (int i = 0; i < items.Length; i++)
    Console.WriteLine(items[i]);
```

Нарисуйте созданные объекты и напишите, что код выводит на экран.

(9) Ниже приведено продолжение этого фрагмента кода:

```
items[0].Foo();
items[1].Goo();
((AA) items[3]).Goo();
items[4].Bar();
BB temp = new BB(1);
temp.Bar();
Console.WriteLine("Temp: " + temp);
temp.Bar(9);
items[2] = temp;
for (int i = 0; i < items.Length; i++)
    Console.WriteLine(items[i]);
```

Напишите, что код выводит на экран.

Желаем успеха!

Авторские права принадлежат Государству Израиль.
Копировать или публиковать можно только
с разрешения Министерства просвещения.

בהצלחה!

זכות היוצרים שמורה למדינת ישראל.
אין להעתיק או לפרסם
אלא ברשות משרד החינוך.