Estado de Israel
Ministerio de Educación
Tipo de Examen: Bachillerato
Fecha de examen: verano 2018
Cuestionario número: 899381
Traducción al español (12)

Ciencias de la Computación

Instrucciones para el examen

א. Duración del examen: tres horas.

ב. Estructura del cuestionario y clave de la evaluación:

Este cuestionario consta de tres partes.

Primera parte -Esta parte consta de tres preguntas, respóndelas según las instrucciones de esta parte.
(1 x 10) + (1 x 15) — 25 puntos

Segunda parte -Esta parte consta de tres preguntas de las que debes responder a dos.
(2x25) — 50 puntos

Tercera parte -Esta parte consta de preguntas que se pueden responder según cuatro orientaciones diferentes.
Responde a una sola de las preguntas según la orientación que estudiaste.
(1x25) — 25 puntos
Total — 100 puntos

ג. Material auxiliar permitido:

Cualquier material auxiliar, salvo calculadoras programables.

ד. Instrucciones especiales:

(1) En la primera y segunda partes, debes escribir todos los programas que te son requeridos en un solo lenguaje: C# o Java.

(2) **Indica en la cobertura externa** del cuadernillo cuál es el **lenguaje** de programación con el que respondes: C# o Java.

(3) **Indica en la cobertura externa** del cuadernillo el nombre de la **orientación según la cual estudiaste**, una de cuatro orientaciones: Sistemas de computación y Assembler, Introducción a la Investigación Operativa, Modelos computacionales, Programación orientada a objetos.

Nota: No se te descontarán puntos en el caso que escribas minúsculas o mayúsculas unas en lugar de otras en el momento de escribir programas.

Escribe solamente en el cuadernillo de examen, y utiliza como páginas de borrador para anotaciones varias (cálculos, títulos. etc.) otras páginas del cuadernillo. Indica con la palabra "טיוטה" cada una de las hojas que utilices como borrador. ¡Toda anotación realizada en hojas que no pertenezcan al cuadernillo del examen puede causar la anulación del mismo!

¡Buena Suerte!

מדינת ישראל
משרד החינוך
סוג הבחינה: בגרות
מועד הבחינה: קיץ תשע"ח, 2018
מספר השאלון: 899381
תרגום לספרדית (12)

מדעי המחשב

הוראות לנבחן

א. משך הבחינה: שלוש שעות.

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

פרק ראשון — בפרק זה שלוש שאלות, ענה על פי ההוראות בפרק.
— (10×1) + (15×1) — 25 נקודות

פרק שני — בפרק זה שלוש שאלות, ומהן עליך לענות על שתיים.
— (25×2) — 50 נקודות

פרק שלישי — בפרק זה שאלות בארבעה מסלולים שונים. ענה על שאלה אחת במסלול שלמדת.—
(25×1) — 25 נקודות
סה"כ — 100 נקודות

ג. חומר עזר מותר בשימוש:

כל חומר עזר, חוץ ממחשב הניתן לתכנות.

ד. הוראות מיוחדות:

(1) את כל התכניות שאתה נדרש לכתוב בשפת מחשב בפרק הראשון והשני כתוב בשפה אחת בלבד — C# או Java.

(2) **רשום על הכריכה החיצונית** של המחברת באיזו **שפה** אתה כותב — C# או Java.

(3) **רשום על הכריכה החיצונית** של המחברת את שם ה**מסלול שלמדת**, אחד מארבעת המסלולים: מערכות מחשב ואסמבלי, מבוא לחקר ביצועים, מודלים חישוביים, תכנות מונחה עצמים.

הערה: בתכניות שאתה כותב לא יורדו לך נקודות, אם תכתוב אות גדולה במקום אות קטנה או להפך.

כתוב במחברת הבחינה בלבד, בעמודים נפרדים, כל מה שברצונך לכתוב **כטיוטה** (ראשי פרקים, חישובים וכדומה). רשום "טיוטה" בראש כל עמוד טיוטה. רישום טיוטות כלשהן על דפים מחוץ למחברת הבחינה עלול לגרום לפסילת הבחינה!

בהצלחה!

Preguntas

Este cuestionario consta de tres partes:

Debes responder a preguntas de las tres partes, según las instrucciones de cada parte.

Primera parte (25 puntos)

Nota: En cada pregunta en la que haga falta realizar una entrada no es necesario revisar su corrección.

Para quienes resuelven en Java: en cada pregunta en la que haga falta realizar una entrada, debes suponer que en el programa se halla escrita la instrucción:

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

Responde a la pregunta 1 obligatoria (10 puntos).

1. Se da la clase **Allnumbers** que posee un atributo: un vector unidimensional `arrayNum` de tipo entero. En el vector hay números positivos mayores que cero, algunos de los cuales son pares y otros son impares.

Escribe un método interno lastOddValue en Java o LastOddValue en C# que devuelva el valor del último número impar del vector.

Por ejemplo, para el vector `arrayNum` de dimensión 6 que se halla a continuación el método devolverá el número 3, que es el valor del último número impar del vector.

	0	1	2	3	4	5
arrayNum	7	5	8	9	**3**	4

Debes suponer que hay por lo menos un número impar.

/continúa en la página 3/

Responde a una de las preguntas 2 - 3 (15 puntos)

2. En el concurso de canciones compiten 40 canciones numeradas de 1- 40. 25 árbitros clasifican las canciones.

Cada árbitro elige las que en su opinión son las tres mejores canciones. A la canción que elige en primer lugar (la mejor) le asigna 7 puntos. A la canción que elige en segundo lugar le asigna 5 puntos y a la que elige en el tercero le asigna 1 punto.

La canción que acumule la mayor suma de puntos asignados por todos los jueces obtendrá el primer puesto.

Para el concurso de canciones se definió una clase **Vote**.

La clase posee tres atributos:

first — número de la canción que eligió el árbitro en primer lugar, de tipo entero.

second — número de la canción que eligió el árbitro en segundo lugar, de tipo entero.

third — número de la canción que eligió el árbitro en tercer lugar, de tipo entero.

Debes suponer que para todos los atributos se han definido en Java métodos get y set**,** y en C# métodos Get y Set.

Escribe un método externo theWinner en Java o TheWinner en C# que reciba un vector de tipo **Vote** con las clasificaciones de los árbitros, e imprima el número de la canción que obtuvo el primer puesto.

Debes suponer que hay una sola canción ganadora.

Debes suponer que los valores del vector son correctos.

/continúa en la página 4/

3. Se da la clase **Time** que tiene dos atributos:
hour – que representa la hora de 0 a 23 incluida, de tipo entero.
minute – que representa el minuto de 0 a 59 incluido, de tipo entero.
Debes suponer que para cada atributo fueron definidos en Java métodos get y set, y en C# métodos Get y Set .

א. Escribe en Java o C# en la clase **Time** un método constructor que reciba valores para **cada** atributo.
Para cada atributo – si el valor que se obtiene no es posible según las definiciónes del atributo, será cambiado por 0.

ב. Se da la clase **Flight** que tiene cuatro atributos: nombre de la compañía aérea– name de tipo cadena de caracteres, nombre del país de destino – destination de tipo cadena de caracteres, código de vuelo – flightCode, de tipo cadena de caracteres, tiempo de vuelo – flightTime de tipo **Time**.
Escribe en Java o C# el título de la clase y sus atributos.

ג. Se da la clase **Airport** que tiene un solo atributo: un vector unidimensional flights de tipo **Flight.**
Debes suponer que para cada atributo de la clase **Flight** fueron definidos en Java métodos get y set, y en C# métodos Get y Set .
En la clase **Airport** se da el método interno booleano isFly en Java o IsFly en C#, cuyo objetivo es estudiar si en el vector flights hay vuelos de la compañía "Sky". El método devuelve `true` si en el vector de vuelos `flights` hay un vuelo de la compañía "Sky" , de lo contrario – el método devuelve `false`.
Debes suponer que todas las celdas del vector son diferentes de null.
El método dado a continuación (isFly en Java y IsFly en C#), **<u>está errado</u>**.

<u>Java</u>

```
public boolean isFly()
{
    for (int i = 0; i < this.flights.length; i++)
    {
        if (this.flights[i].getName().equals ("Sky"))
            return true;
        return false;
    }
}
```

<u>C#</u>

```
public bool IsFly()
{
    for (int i = 0; i < this.flights.Length; i++)
    {
        if (this.flights[i].GetName() == "Sky")
            return true;
        return false;
    }
}
```

(Presta atención los subapartados del apartado ג se hallan en la próxima página.)

/continúa en la página 5

(1) Escribe qué es lo que devolverá el método **erróneo**, dado para un vector flights de dimensión 4 , cuyos valores de atributo name de los objetos en el vector son:

En el primer lugar: "Cloud" ,

En el segundo lugar: "Air" ,

En el tercer lugar: "Sky" ,

En el cuarto lugar "Travel" .

(2) Explica cuál es el error del método.

(3) Corrige el método para que realice lo que se le exige, y copia en tu cuadernillo el método corregido.

.

/continúa en la página 6/

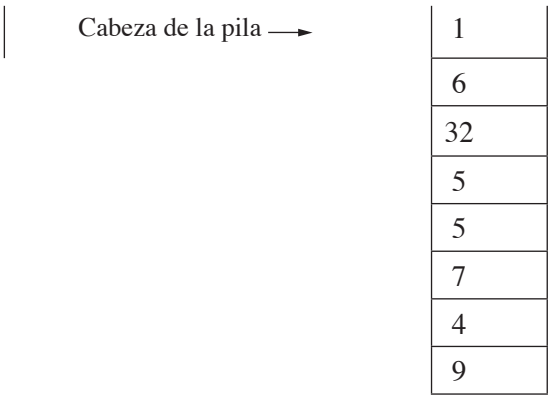
Segunda parte (50 puntos)

Presta atención: en toda pregunta en la que se requiera implementación, puedes utilizar métodos de las clases cola, pila, árbol binario y eslabón, [תור, מחסנית, עץ בינרי וחוליה] sin implementarlos. Si utilizas métodos adicionales, deberás implementarlos.

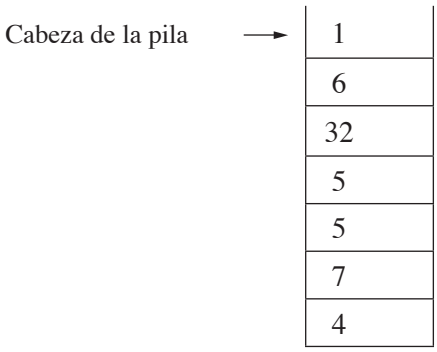
Responde a dos de las preguntas 4– 6 (cada pregunta – 25 puntos).

4. **א**. Escribe un método externo lastAndRemove en Java o LastAndRemove en C# , que recibe una pila de tipo entero, borra el último término de la pila y devuelve su valor .
Al finalizar el método los demás términos de la pila permanecen sin cambio.
Debes suponer que la pila no es vacía.

Por ejemplo, para la pila siguiente:



Después de que el método devuelve el valor 9 , la pila se ve así:



/continúa en la página 7/

ב. Se da la clase:

TwoItems	
Atributos:	int num1 int num2
Método constructor:	TwoItems (int number1, int number2)

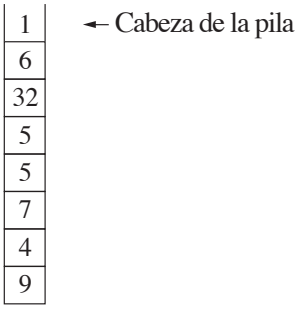
Debes suponer que para cada atributo fueron definidos en Java métodos get y set, y en C# métodos Get y Set .

Escribe un método externo stackTwoItems en Java o StackTwoItems en C# , que reciba una pila stk1 no vacía, de tipo entero y de magnitud par, y devuelva una pila de tipo TwoItems.

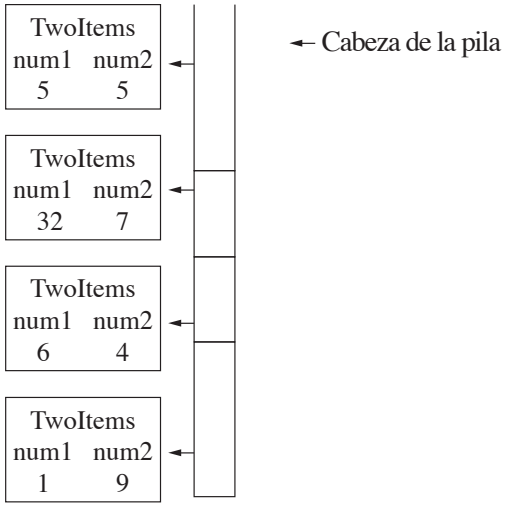
El término inferior de la pila devuelta contendrá la derivación a la aparición de TwoItems cuyo atributo num1 es el término que se halla en la cabeza de la pila stk1 , y su atributo num2 es el término inferior de la pila stk1 .

El término que se halla por encima del término inferior en la pila devuelta contendrá una derivación a la aparición de TwoItems cuyo atributo num1 es el término que se halla debajo del término que está en la cabeza de la pila stk1 , y su atributo num2 es el término que se halla por encima del término inferior de la pila stk1, y así sucesivamente, de modo que el término de la cabeza de la pila devuelta contenga la derivación a la aparición de TwoItems cuyos atributos num1 y num2 son dos términos adyacentes que se hallan en medio de la pila stk1 .

Por ejemplo, para la pila stk1 que sigue:



Después del método se devolverá la pila:



Debes ayudarte con el método que escribiste en el apartado א.

Nota: no es necesario guardar el contenido original de la pila stk1 .

/continúa en la página 8/

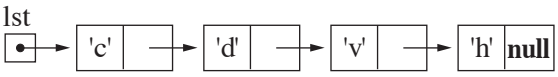
5. <u>Presta atención</u>: <u>Esta pregunta tiene una versión en Java en la página 8 y una versión en C# en la página 9</u>.

<u>Para los que resuelven en Java</u>

Se da el método sod1:

```
public static Node <Character> sod1(Node <Character> lst, char ch)
{
    if (lst == null)
        return null;
    if (lst.getValue() == ch)
        return lst;
    return sod1(lst.getNext(), ch);
 }
```

א. **(1)** Realiza el seguimiento del método y escribe qué devolverá para ch = 'v' y la derivación lst para la cadena de eslabones de caracteres.



(2) ¿Cuál es la finalidad del método sod1?

(3) ¿Cuál es la complejidad del tiempo de ejecución del método sod1? Justifica tu respuesta.

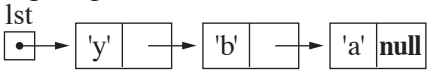
ב. Se da el método sod2 :

```
public static boolean sod2(Node <Character> lst)
{
    if (sod1(lst,'a') != null && sod1(lst,'b') != null)
        return true;
    return false;
}
```

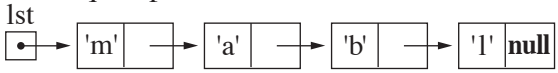
¿Cuál es la finalidad del método sod2?

ג. Escribe un método booleano que reciba una derivación a la cadena de eslabones de caracteres y devuelve true si aparecen en ella dos eslabones adyacentes cuyos valores son 'a' 'b' <u>o</u> 'b' 'a' . En caso contrario — el método devuelve false .

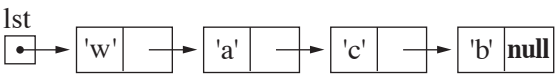
Ejemplo de una cadena de eslabones en la que aparece 'b' 'a' en continuidad:



Ejemplo de una cadena de eslabones en la que aparece 'a' 'b' en continuidad:



Ejemplo de una cadena de eslabones en la que <u>no</u> aparecen ni 'a' 'b' ni 'b' 'a' en continuidad:



Debes utilizar el método sod1 .

Debes suponer que todos los caracteres son diferentes entre sí.

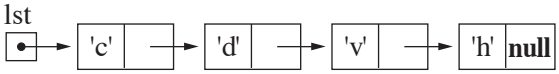
/continúa en la página 9/

Para los que resuelven en C#

Se da el método Sod1:

```
public static Node <char> Sod1(Node <char> lst, char ch)
{
    if (lst == null)
        return null;
    if (lst.GetValue() == ch)
        return lst;
    return Sod1(lst.GetNext(), ch);
}
```

א. **(1)** Realiza el seguimiento del método y escribe qué devolverá para ch = 'v' y la derivación lst para encadenar los eslabones de caracteres.



(2) ¿Cuál es la finalidad del método Sod1?

(3) ¿Cuál es la complejidad del tiempo de ejecución del método Sod1? Justifica tu respuesta.

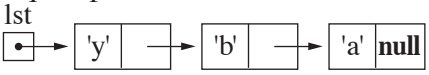
ב. Se da el método Sod2 :

```
public static bool Sod2(Node <char> lst)
{
    if (Sod1(lst,'a') != null && Sod1(lst,'b') != null)
        return true;
     return false;
}
```

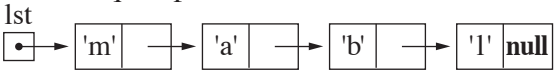
¿Cuál es la finalidad del método Sod2?

ג. Escribe un método booleano que reciba una derivación a la cadena de eslabones de caracteres y devuelve true si aparecen en ella dos eslabones adyacentes cuyo valores son 'a' 'b' o 'b' 'a' . En caso contrario — el método devuelve false .

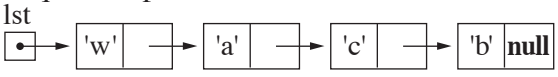
Ejemplo de una cadena de eslabones en la que aparece 'b' 'a' en continuidad:



Ejemplo de una cadena de eslabones en la que aparece 'a' 'b' en continuidad:



Ejemplo de una cadena de eslabones en la que no aparecen ni 'a' 'b' ni 'b' 'a' en continuidad:



Debes utilizar el método Sod1 .

Debes suponer que todos los caracteres son diferentes entre sí.

/continúa en la página 10/

6. Se da el método:

En Java :

public static boolean lessThanTree (BinNode <Integer> t, int x)

En C # :

public static bool LessThanTree (BinNode <int> t, int x)

El método devuelve true si x es menor que todos los valores en el árbol t . En caso contrario — el método devuelve false .

La complejidad de tiempo de ejecución del método es O(n) . n representa el número de nodos del árbol t .

א. Escribe un método externo treeLessThanTree en Java o TreeLessThanTree en C# , que reciba dos árboles binarios t1 y t2 de valores enteros. Se sabe que en t2 existe por lo menos un nodo. El método devuelve true si todos los valores de árbol t1 son menores que todos los valores del árbol t2 , en caso contrario — el método devuelve false .

Si t1 es null — el método devuelve true .

Se puede utilizar el método dado sin implementarlo. Si utilizas otros métodos debes implementarlos.

ב. ¿Cuál es la complejidad del tiempo de operación del método que escribiste en el apartado א? Justifica tu respuesta.

/continúa en la página 11/

Tercera parte (25 puntos)

Esta parte consta de preguntas según cuatro orientaciones:

Sistemas de computación y Assembler, págs. 11-13

Introducción a la investigación operativa, págs. 14-17

Modelos computacionales, págs. 18-19.

Programación orientada a objetos en Java, págs. 20-23.

Programación orientada a objetos en C#, págs. 24-27.

Responde a una pregunta de la orientación que estudiaste.

Sistemas de computación y Assembler

Si estudiaste en esta orientación, responde a una de las preguntas 7-8
(cada pregunta – 25 puntos).

7. En esta pregunta hay dos apartados, א-ב, que no tienen relación entre sí. Responde a ambos.

א. Te presentamos un segmento de programa en Assembler.
Debes suponer que los números son faltos de signo (Unsigned).

```
        CMP  AH , AL
        JAE  AA
        XCHG AH, AL
AA:     CMP AL, AH
        JAE CC
        INC AL
        DEC AH
        JMP AA
CC:     MOV SI, 100H
        MOV [SI], AL
```

(1) El contenido del acumulador AX es 0109H . Sigue mediante una tabla de seguimiento la ejecución del segmento de programa, y escribe qué se obtendrá en la celda de dirección 100H después de la ejecución.

(2) ¿Qué ejecuta el segmento de programa para los números impares en los acumuladores AH, AL? . En tu respuesta, refiérete también al contenido de los acumuladores AH, AL antes de la ejecución del segmento de programa.

/continúa en la página 12/

ב. (Sin relación con el apartado א.)
A continuación seis proposiciones. Para cada una de ellas determina si es cierta o no. Si la proposición no es cierta explica por qué.

(1) Los dos segmentos 1 y 2 que siguen ejecutan lo mismo.

Segmento 2	Segmento 1
SHR AX, 1	SHL AX, 1
SHL AX, 1	SHR AX, 1

(2) Los dos segmentos 1 y 2 que siguen ejecutan lo mismo.

Segmento 2	Segmento 1
SHR AX, 1	AND AX, 0FEh
SHL AX, 1	

(3) Los dos segmentos 1 y 2 que siguen ejecutan lo mismo.

Segmento 2	Segmento 1
XCHG AX, BX	PUSH AX
	PUSH BX
	POP AX
	POP BX

(4) Los dos segmentos 1 y 2 que siguen ejecutan lo mismo.

Segmento 2	Segmento 1
CMP AX, 0	SUB AX, 0
JNZ A1	JNE A1

.

(5) En el segmento de datos se definió un vector:
ARR DW 50 dup (?)
Debes suponer que en el acumulador BX se almacenó la dirección de un término determinado en el vector ARR.
Este segmento de programa acumula en BX el índice de dicho término.

```
LEA SI, ARR
SUB BX,SI
INC BX
```

(6) El valor del número 11101011 que está almacenado según el método : completa a 2 en 8 bits es (—21) en base 10.

/continúa en la página 13/

8. En el segmento de datos que se halla a continuación se definieron los datos:

ARR DB 100 DUP (?)

REZ DB ?

Debes suponer que los números poseen signos (Signed).

Debes suponer que todos los números del vector son diferentes entre sí.

Un vector se llama "גלי" (Gali) si cumple con las siguientes condiciones:
Su primer término es menor que el segundo, el segundo término es mayor que el tercero, el tercer término es menor que el cuarto, y el cuarto término es mayor que el quinto, y así sucesivamente.
Ejemplo de un vector de tipo "גלי":

0	1	2	3	4
−2	8	6	17	−7

Explicación del ejemplo: El primer término (−2) es menor que el segundo (8), el segundo término (8) es mayor que el tercero (6), el tercer término (6) es menor que el cuarto (17) el cuarto término (17) es mayor que el quinto (−7).

Escribe un segmento de programa que analice si el vector ARR es un vector "גלי". En caso afirmativo, debes introducir en la variable REZ el valor 1. En caso contrario debes introducir en la variable REZ el valor 0.

/continúa en la página 14/

Introducción a la Investigación operativa [חקר ביצועים]

Si estudiaste en esta orientación, responde a una de las preguntas 9-10 (25 puntos).

9. Se da un problema de programación lineal:

$\max\{z = 4x_1 + 6x_2\}$

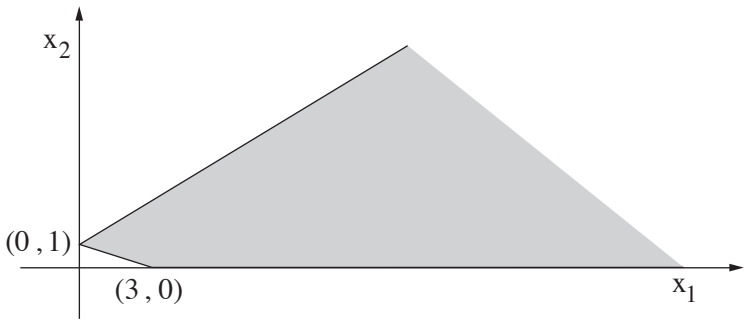
Sometido a las restricciones siguientes:

$$(1) \quad -5x_1 + 4x_2 \leq 4$$

$$(2) \quad x_1 + 3x_2 \geq 3$$

$$(3) \quad x_2 \geq 0$$

A continuación se halla el dibujo del intervalo de las soluciones posibles del problema dado.



Cada uno de los apartados א-ו que se encuentran en la página siguiente se refieren al problema de programación lineal dado.

Los apartados א-ו son independientes entre sí. Responde a todos los apartados.

Se dan cuatro proposiciones i-iv, y para cada uno de los apartados א-ו de la página siguiente sólo una de las proposiciones es cierta.

i Existe una única solución óptima.

ii Existe un número infinito de soluciones óptimas.

iii La solución óptima no está acotada.

iv No existe solución óptima.

/continúa en la página 15/

En cada uno de los apartados א-ו determina cuál de las proposicines i-iv es verdadera. Señala el apartado, copia la proposición verdadera en tu cuadernillo y fundamenta tu afirmación.

- Si elegiste la proposición i en un apartado cualquiera, debes encontrar la solución óptima única y el valor de la función objetivo en dicha solución.
- Si elegiste la proposición ii en un apartado cualquiera, debes anotar la solución óptima general del problema, y el valor de la función objetivo en el intervalo de soluciones óptimas.

א. ¿Qué proposición es cierta para el problema de programación lineal dado al comienzo de la pregunta? Justifica tu respuesta.

ב. Se cambia solamente la función objetivo del problema dado al comienzo de la pregunta a: $\min\{z = 4x_1 + 6x_2\}$.

¿Qué proposición es cierta después del cambio? Justifica tu respuesta.

ג. Se cambia solamente la función objetivo del problema dado al comienzo de la pregunta a: $\max\{z = 2x_1 + 6x_2\}$.

¿Qué proposición es cierta después del cambio? Justifica tu respuesta.

ד. Se cambia solamente la función objetivo del problema dado al comienzo de la pregunta a: $\min\{z = 2x_1 + 6x_2\}$.

¿Qué proposición es cierta después del cambio? Justifica tu respuesta.

ה. Se agrega una restricción al problema dado al comienzo de la pregunta, y es: $x_1 + x_2 \leq 1$.

¿Qué proposición es cierta después de agregar la restricción? Justifica tu respuesta.

ו. Se agrega una restricción al problema dado al comienzo de la pregunta, y es: $x_2 \leq -x_1$.

¿Qué proposición es cierta después de agregar la restricción? Justifica tu respuesta.

/continúa en la página 16/

10. En esta pregunta hay dos apartados א-ב independientes uno del otro.
Debes responder a ambos.

א. G = (V , E) es un grafo **orientado** representado por la siguiente matriz de adyacencias:

	a	**b**	**c**	**d**	**e**
a	0	0	1	1	0
b	1	0	0	1	0
c	0	0	0	0	1
d	0	1	0	0	0
e	1	0	0	0	0

(1) Dibuja el grafo G representado por la matriz de adyacencias.

(2) Halla los componentes de conexidad fuerte (Strong Connected Components, c.c.f. - רק"חים) que hay en el grafo dado.
Para cada c.c.f. que halles, escribe el conjunto de sus vértices.

(3) Determina el número máximo de arcos que es posible retirar del grafo dado, y que el grafo continúe teniendo el mismo número de c.c.f.? que hallaste en el subapartado א (2)
¿Cuál es el arco o los arcos que hay que retirar?

ב. (Sin relación con el apartado א.)

(1) En la tabla que está a continuación se da un problema de transporte y una parte de una solución básica posible: $x_{11} = 9$, $x_{12} = 1$.

Orígenes	**Destinos**			**Oferta**
	1	2	3	
1	1 (9)	5 (1)	7	10
2	1	8	4	11
3	5	2	8	10
Demanda	9	12	10	

Copia la tabla en tu cuadernillo y completa en ella los valores de la solución según el método de la esquina noroeste.

/continúa en la página 17/

(2) En la tabla que está a continuación se da una parte de una solución básica posible de un problema de transporte y se dan los valores de $u_1 , u_2 , u_3 , v_1 , v_2 , v_3$.

Orígenes	**Destinos**			**Oferta**	u_i
	1	2	3		
1	3 (20)	5	7	20	1
2	2	8 (10)	14	10	0
3	2	6	8 (10)	15	-2
Demanda	20	15	10		
v_j	2	8	10		

Copia la tabla en tu cuadernillo y complétala teniendo en cuenta los valores de <u>todos</u> los u_i y de <u>todos</u> los v_j de modo que se obtenga una solución <u>básica</u> posible.

(3) En la tabla que está a continuación se da una solución básica posible de un problema de transporte y se dan los valores de $u_1 , u_2 , u_3 , v_1 , v_2 , v_3$.

Orígenes	**Destinos**			**Oferta**	u_i
	1	2	3		
1	34	15 (15)	17 (3)	18	0
2	10 (10)	8 (0)	4	10	-7
3	25	18	18 (10)	10	1
Demanda	10	15	13		
v_j	17	15	17		

¿Se trata de una solución óptima? Justifica tu respuesta.

/continúa en la página 18/

Modelos computacionales

Si estudiaste en esta orientación, responde a una de las preguntas 11-12 (25 puntos).

11. En esta pregunta hay dos apartados א - ב independientes uno del otro. Responde a ambos apartados.

א. A continuación la definición: Se llama **antecedente [רישא]** de una palabra x a toda palabra que se obtiene a partir del retiro de un número cualquiera de caracteres del final de la palabra x, incluyendo la palabra vacía y la propia palabra x.

Por ejemplo: Para la palabra $x = abcbad$ todos los antecedentes de la palabra x son:

$\varepsilon, a, ab, abc, abcb, abcba, abcbad$

Se da el lenguaje L sobre el alfabeto $\Sigma = \{a,b,c,d\}$.

L es una colección de palabras en la que cada una de ellas para todo **antecedente** de la palabra – la diferencia entre el número de veces que aparece el carácter c y el número de veces que aparece el carácter d es mayor o igual que 0, y menor o igual que 3:

$$0 \le \#_c(w) - \#_d(w) \le 3$$

$\#_c(w)$ indica el número de veces que aparece c en la palabra w .

$\#_d(w)$ indica el número de veces que aparece d en la palabra w .

Ejemplos de palabras que pertenecen al lenguaje L :

accbdcacab , bacaabdbcb , abba , cdcdcd , abcbadb

Ejemplos de palabras que no pertenecen al lenguaje L :

daac — porque existe el antecedente d , y en él: $\#_c(w) - \#_d(w) = -1 < 0$

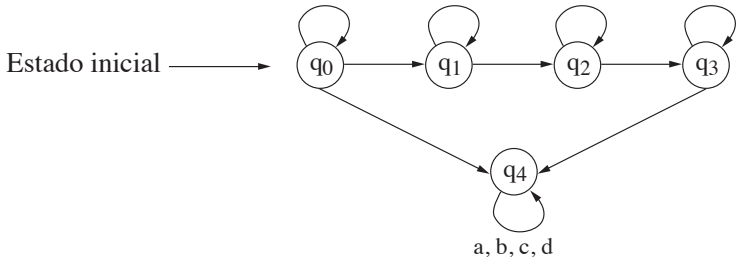
cddc — porque existe el antecedente cdd , y en él: $\#_c(w) - \#_d(w) = -1 < 0$

accbdcacacd — porque existe el antecedente accbdcacac , y en él: $\#_c(w) - \#_d(w) = 4 > 3$

(Presta atención: la continuación de la pregunta se halla en la página siguiente)

/continúa en la página 19/

A continuación se halla un dibujo parcial de un autómata finito **determinístico** que acepta el lenguaje L. En el dibujo faltan transiciones, signos de entrada y estados de aceptación. En el dibujo se incluyen todos los estados del autómata.



Copia en tu cuadernillo el dibujo y complétalo de modo que el autómata sea **determinístico** y admita el lenguaje L.
Debes completar las transiciones faltantes y los signos de entrada faltantes, y señalar **todos los estados de aceptación**.
Presta atención: No hay que agregar estados, y no hay que quitar ni estados ni transiciones al autómata.

ב. (Sin relación con el apartado א.)

Σ^* es una colección de palabras sobre el alfabeto Σ, incluyendo la palabra vacía.

Se dan dos lenguajes L_2 , L_1 sobre el alfabeto Σ.

$L_1 = \Sigma^*$ y L_2 es un lenguaje no regular.

Definiremos: $L_3 = L_2 \cap \overline{L_1}$.

(1) ¿Cuál es el lenguaje $\overline{L_1}$?

(2) ¿Es el lenguaje L_3 regular? Justifica tu respuesta.

12. En esta pregunta hay dos apartados א - ב independientes uno del otro. Responde a ambos apartados.

א. Se da el alfabeto $\Sigma = \{a,b\}$. Construye un autómata de pila para el lenguaje:

$$L = \{a^2 b^k a^n \mid n > 0 \quad , \quad k > 0 \quad , \quad n < k\}$$

ב. (Sin relación con el apartado א.)

Construye una máquina de Turing que acepte como entrada el número x en representación unaria, y calcula el valor de la función que sigue.

$$f(x) = \begin{cases} x+1 & x < 2 \\ x-1 & x \geq 2 \end{cases}$$

/continúa en la página 20/

Programación orientada a objetos

Si estudiaste en esta orientación y escribes en Java, responde a una de las preguntas 13-14. (25 puntos)

13. A continuación un proyecto en el que se definieron las clases A, B, C, D, E. En el proyecto se implementó el método f() en dos clases.
El segmento de código que se halla a continuación es **correcto**.

```
A a1 = new A();
A e1 = new E();
E c1 = new C();
C b1 = new B();
C d1 = new D();
```

Datos:
La instrucción B d2 = new D(); provoca un error de compilación (קומפילציה).
La instrucción a1.f(); provoca un error de compilación (קומפילציה).
La instrucción ((E)e1).f(); es correcta e imprime "bye-bye" .
La instrucción ((B)b1).f(); es correcta e imprime "hello" .
La instrucción ((D)b1).f(); provoca un error de tiempo de ejecución.

א. Dibuja el árbol de herencia de todas las clases, y señala en cuáles dos clases se implementó el método f() .

ב. Se da la clase Z :

```
public class Z
{
    public void g(){}
}
```

(1) Agrega la clase Z al árbol de herencia que dibujaste en el apartado א de modo que la instrucción que sigue sea correcta:

```
Z x = new A();
```

(2) El segmento de código que se halla a continuación es **correcto**.

```
A a2 = new A();
Z z1 = new Z();
Z a3 = new A();
```

Para cada una de las instrucciones i-v que se hallan a continuación indica si la instrucción es o no correcta.
Si la instrucción no es correcta, explica por qué.
Los apartados i-v son **independientes** unos de otros.

```
i    a2 = z1;
ii   a3 = z1;
iii  ((A) z1) .g();
iv   a2.g();
v    ((A)a3).g();
```

/continúa en la página 21/

ג. Se da la clase Y :

```
public class Y
{
    public void m(){}
}
```

(1) Agrega la clase Y al árbol de herencia que dibujaste en el apartado א de modo que la instrucción que sigue sea correcta:

(Sin relación con los datos del apartado ב).

```
C f = new Y();
```

(2) El segmento de código que se halla a continuación es **correcto**.

```
A a2 = new A();
Y y1 = new Y();
C y2 = new Y();
```

Para cada una de las instrucciones i-v que se hallan a continuación indica si la instrucción es o no correcta.
Si la instrucción no es correcta, explica por qué.

Los apartados i-v son **independientes** unos de otros.

i `a2 = y1;`

ii `y2 = y1;`

iii `((A)y1).m();`

iv `y2.m();`

v `((A)y2).m();`

/continúa en la página 22/

14. El pago del impuesto inmobiliario (Property tax) es un pago a los municipios por el área de las viviendas y el suelo adicional que se halla en propiedad de los habitantes. El precio del impuesto es de 10 sheqalim por m^2 de vivienda y 0.5 sheqalim por m^2 de suelo adicional.

Por ejemplo, el habitante que posee un área de vivienda de 100 m^2 y un área de suelo adicional de 500 m^2 paga impuesto inmobiliario de $100 \cdot 10 + 500 \cdot 0.5 = 1250$.

Los habitantes fueron divididos en tres grupos:

* El habitante de la ciudad paga impuestos inmobiliarios por el área de las viviendas y del suelo que se halla en su propiedad y recibe una bonificación de 250 sheqalim para el pago del impuesto.
* El antiguo habitante — habitante de la ciudad de 60 años y más — paga impuestos inmobiliarios como el habitante de la ciudad, y además se recibe un descuento de acuerdo a su edad: 1% de descuento por cada año de vida después de los 60. El descuento se calcula después de la deducción de la bonificación sobre la suma original.
* El habitante del campo paga impuestos inmobiliarios por el área de las viviendas y del suelo que se halla en su propiedad y recibe un descuento del 10% . **El habitante del campo de 60 años y más, no es acreedor de un descuento que dependa de la edad**.

Si después del descuento y de la deducción de la bonificación la suma del impuesto inmobiliario a pagar es negativo, el habitante está eximido del pago (paga 0 sheqalim).

Se decidió desarrollar un programa para calcular los impuestos inmobiliarios y para su cobro de los habitantes, y para ello se definió la interface siguiente:

```
interface IData
{
    public String getName(); // Devuelve el nombre del habitante
    public double getPropertyTax(); // Devuelve el importe del impuesto inmobiliario
{
```

En los apartados א-ב que se hallan a continuación debes definir las clases según los principios de la programación orientada a objetos y de acuerdo a las exigencias:

* Los datos para el cálculo de los impuestos inmobiliarios (pago de acuerdo al área de la vivienda en m^2, el pago de acuerdo al área del suelo en m^2 , el importe del descuento en porcentaje, la bonificación en sheqalim) deben definirse una sola vez como constantes.
* Todas las propiedades que no son constantes defínelas como privadas (private).

/continúa en la página 23/

א. Define la clase Resident (que representa habitante), que implementa la interface IData , y escribe en ella los atributos de la clase (no hace falta escribir un método constructor).

ב. Define tres clases para cada una de las clases de habitantes: CityResident — habitante de la ciudad, SeniorCityResident — antiguo habitante de la ciudad, VillageResident — habitante del campo. Escribe los atributos de las clases y el método getPropertyTax() que permita calcular el pago del impuesto inmobiliario a cada una de las tres clases de habitante (no hace falta escribir un método constructor).

ג. Se da un cuerpo de método que recibe un vector a de datos de los habitantes, e imprime para cada antiguo habitante de la ciudad su nombre y la suma del impuesto inmobiliario que se le exige pagar.

```
for (int i = 0;  i < a.length; i++)
{
    if (a[i] instanceof SeniorCityResident)
    {
        System.out.println(a[i].getName()+" "+a[i].getPropertyTax());
    }
}
```

Para cada uno de los títulos del método (1)-(3) que se hallan a continuación, señala si el método es correcto o no es correcto. Si el método no fuera correcto – explica por qué, corrige el método, y copia el método corregido en tu cuadernillo.

(1) public static void print(Object [] a)

(2) public static void print(IData [] a)

(3) public static void print(Resident [] a)

/continúa en la página 24/

Programación orientada a objetos:

Si estudiaste en esta orientación y escribes en C#, responde a una de las preguntas 15-16. (25 puntos)

15. A continuación un proyecto en el que se definieron las clases A, B, C, D, E. En el proyecto se implementó el método F() en dos clases.

El segmento de código que se halla a continuación es **correcto**.

```
A a1 = new A();
A e1 = new E();
E c1 = new C();
C b1 = new B();
C d1 = new D();
```

Datos:

La instrucción B d2 = new D(); provoca un error de compilación (קומפילציה).

La instrucción a1.F(); provoca un error de compilación (קומפילציה).

La instrucción ((E)e1).F(); es correcta e imprime "bye-bye" .

La instrucción ((B)b1).F(); es correcta e imprime "hello" .

La instrucción ((D)b1).F(); provoca un error de tiempo de ejecución.

א. Dibuja el árbol de herencia de todas las clases, y señala en cuáles dos clases se implementó el método F() .

ב. Se da la clase Z :

```
public class Z
{
    public void G(){}
}
```

(1) Agrega la clase Z al árbol de herencia que dibujaste en el apartado א de modo que la instrucción que sigue sea correcta:

```
Z x = new A();
```

(2) El segmento de código que se halla a continuación es **correcto**.

```
A a2 = new A();
Z z1 = new Z();
Z a3 = new A();
```

Para cada una de las instrucciones i-v que se hallan a continuación indica si la instrucción es o no correcta.

Si la instrucción no es correcta, explica por qué.

Los apartados i-v son **independientes** unos de otros.

```
i    a2 = z1;
ii   a3 = z1;
iii  ((A)z1).G()
iv   a2.G();
v    ((A)a3).G();
```

/continúa en la página 25/

ג. Se da la clase Y :

```
public class Y
{
    public void M(){}
}
```

(1) Agrega la clase Y al árbol de herencia que dibujaste en el apartado א de modo que la instrucción que sigue sea correcta:

(Sin relación con los datos del apartado ב).

```
C f = new Y();
```

(2) El segmento de código que se halla a continuación es **correcto**.

```
A a2 = new A();
Y y1 = new Y();
C y2 = new Y();
```

Para cada una de las instrucciones i-v que se hallan a continuación indica si la instrucción es o no correcta.
Si la instrucción no es correcta, explica por qué.

Los apartados i-v son **independientes** unos de otros.

i `a2 = y1;`

ii `y2 = y1;`

iii `((A)y1).M();`

iv `y2.M();`

v `((A)y2).M();`

/continúa en la página 26/

16. El pago del impuesto inmobiliario (Property tax) es un pago a los municipios por el área de las viviendas y el suelo adicional que se halla en propiedad de los habitantes. El precio del impuesto es de 10 sheqalim por m^2 de vivienda y 0.5 sheqalim por m^2 de suelo adicional.

Por ejemplo, el habitante que posee un área de vivienda de 100 m^2 y un área de suelo adicional de 500 m^2 paga impuesto inmobiliario de $100 \cdot 10 + 500 \cdot 0.5 = 1250$.

Los habitantes fueron divididos en tres grupos:

* El habitante de la ciudad paga impuestos inmobiliarios por el área de las viviendas y del suelo que se halla en su propiedad y recibe una bonificación de 250 sheqalim para el pago del impuesto.
* El antiguo habitante — habitante de la ciudad de 60 años y más — paga impuestos inmobiliarios como el habitante de la ciudad, y además se recibe un descuento de acuerdo a su edad: 1% de descuento por cada año de vida después de los 60. El descuento se calcula después de la deducción de la bonificación sobre la suma original.
* El habitante del campo paga impuestos inmobiliarios por el área de las viviendas y del suelo que se halla en su propiedad y recibe un descuento del 10% . **El habitante del campo de 60 años y más, no es acreedor de un descuento que dependa de la edad**.

Si después del descuento y de la deducción de la bonificación la suma del impuesto inmobiliario a pagar es negativo, el habitante está eximido del pago (paga 0 sheqalim).

Se decidió desarrollar un programa para calcular los impuestos inmobiliarios y para su cobro de los habitantes, y para ello se definió la interface siguiente:

```
interface IData
{
    string GetName(); // Devuelve el nombre del habitante
    double GetPropertyTax(); // Devuelve el importe del impuesto inmobiliario
{
```

En los apartados ב-א que se hallan a continuación debes definir las clases según los principios de la programación orientada a objetos y de acuerdo a las exigencias:

* Los datos para el cálculo de los impuestos inmobiliarios (pago de acuerdo al área de la vivienda en m^2, el pago de acuerdo al área del suelo en m^2 , el importe del descuento en porcentaje, la bonificación en sheqalim) deben definirse una sola vez como constantes.
* Todas las propiedades que no son constantes defínelas como privadas (private).

/continúa en la página 27/

א. Define la clase Resident (que representa habitante), que implementa la interface IData , y escribe en ella los atributos de la clase (no hace falta escribir un método constructor).

ב. Define tres clases para cada una de las clases de habitantes: CityResident — habitante de la ciudad, SeniorCityResident — antiguo habitante de la ciudad, VillageResident — habitante del campo. Escribe los atributos de las clases y el método GetPropertyTax() que permita calcular el pago del impuesto inmobiliario a cada una de las tres clases de habitante (no hace falta escribir un método constructor).

ג. Se da un cuerpo de método que recibe un vector a de datos de los habitantes, e imprime para cada antiguo habitante de la ciudad su nombre y la suma del impuesto inmobiliario que se le exige pagar.

```
for ( int i = 0; i < a.Length; i++)
{
    if (a[i] is SeniorCityResident)
    {
        Console.WriteLine (a[i].GetName()+" "+a[i].GetPropertyTax());
    }
}
```

Para cada uno de los títulos del método (1)-(3) que se hallan a continuación, señala si el método es correcto o no es correcto. Si el método no fuera correcto – explica por qué, corrige el método, y copia el método corregido en tu cuadernillo.

(1) public static void Print(Object [] a)

(2) public static void Print(IData [] a)

(3) public static void Print(Resident [] a)

¡Buena Suerte!

בהצלחה!