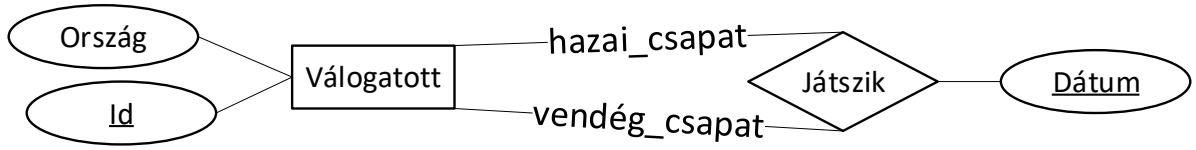
Informatika 2 vizsga 2018.06.19.	
A nevet <u>NYOMTATOTT, NAGYBETŰVEL</u> írja, nem aláírást kérünk. *A rendelkezésre álló idő 90 perc.* *Az elégséges szint 45% (31 pont), de <u>feladatcsoportonként 20% elérése szükséges</u>.*	
NÉV:	NEPTUN:

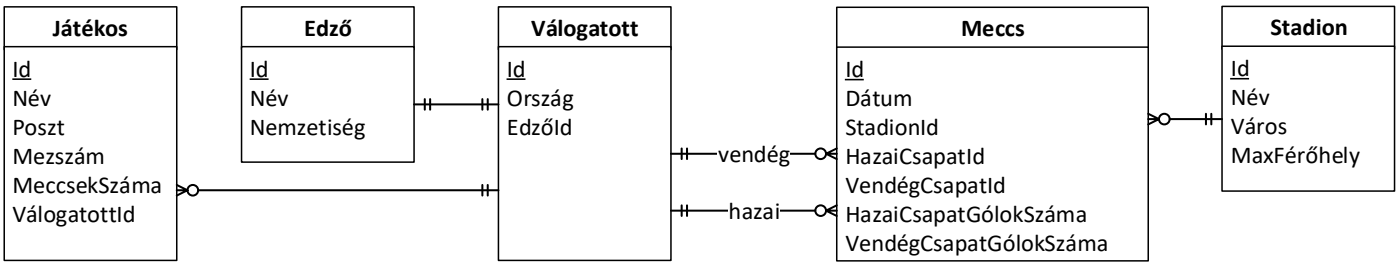
Adatbázisok		Hálózatok		Nyelvek	
1 [10]		4 [10]		7 [4]	
2 [4]		5 [8]		8 [8]	
3 [11]		6 [6]		9 [9]	
∑ [25]		∑ [24]		∑ [21]	
∑ [70]				ZH+HF:	
Összesen:				Jegy:	

1. Feladat Adott az alábbi ábrán látható Entitás-Relációs diagram, amelynek a segítségével a foci vb meccseinek időpontjait szeretnénk eltárolni. Írjon SQL utasításokat, amelyek a diagramnak és a kiegészítéseknek megfelelő táblákat létrehozzák! Az országok neveit kötelező megadni és két azonos nevű ország nem megengedett. [**10 pont**]




```
CREATE TABLE Válogatott(
      Ország NVARCHAR NOT NULL UNIQUE,      -- 2 pont
      Id INT PRIMARY KEY                    -- 1 pont
);
CREATE TABLE Játszik(                                          -- 1 pont
      HazaiCsapat INT,                                         -- 1 pont
      VendégCsapat INT,                                        -- 1 pont
      Dátum DateTime,                                          -- 1 pont
      PRIMARY KEY(HazaiCsapat, VendégCsapat, Dátum),           -- 1 pont
      FOREIGN KEY (HazaiCsapat) REFERENCES Válogatott(Id),     -- 1 pont
      FOREIGN KEY (Vendég) REFERENCES Válogatott(Id),          -- 1 pont
);
```

Adott az alábbi ábrán látható adatbázis séma.



2. Feladat Hány gólt lőtt Németország összesen a vb meccsein hazai csapatként? [**4 pont**]

```
SELECT SUM(HazaiCsapatGólokSzáma)                              -- 1 pont
FROM Meccs
      INNER JOIN Válogatott ON HazaiCsapatId = Válogatott.Id   -- 2 pont
WHERE Ország = 'Németország'                                   -- 1 pont
```

3. Feladat Hány meccset nyert meg összesen Németország a vb-n az egyes stadionokban? Az eredményben a stadionok nevei, mellette a megnyert meccsek száma szerepeljen! [**11 pont**]

```
SELECT s.Név, COUNT(m.Id)                                                    -- 2 pont
FROM Meccs m                                                                 -- 1 pont
      INNER JOIN Válogatott h ON h.Id = m.HazaCsapatId                       -- 1 pont
      INNER JOIN Válogatott v on v.Id = m.VendégCsapatId                     -- 1 pont
      INNER JOIN Stadion s ON s.Id = m.StadionId                             -- 1 pont
WHERE
      HazaiCsapatGólokSzáma > VendégCsapatGólokSzáma AND h.Ország = 'Németország' OR -- 2 pont
      azaiCsapatGólokSzáma < VendégCsapatGólokSzáma AND v.Ország = 'Németország'     -- 2 pont
GROUP BY s.id                                                                -- 1 pont
```

4. Feladat Ismertesse az Internet működését leíró rétegzett architektúrát! Milyen rétegek vannak és mi az egyes rétegek feladata? [**10 pont**]

Réteg neve	**Feladata**
Alkalmazási réteg	Tartalmazza az elosztott alkalmazásokat, az elosztott alkalmazás komponensek egymásnak üzeneteteket küldenek.
Szállítási réteg	Üzenetátvitel a hálózat két pontján lévő eszközökön futó alkalmazás folyamatok között.
Hálózati réteg	Üzenetátvitel a hálózat két pontján lévő eszközök között.
Adatkapcsolati réteg	Üzenetátvitel két szomszédos hálózati csomópont között.
Fizikai réteg	Az üzenetátvitel fizikai megvalósítása.

5. Feladat Ismertesse a CSMA/CA protokoll működését! Milyen esetekben van rá szükség? Milyen opcionális kiegészítése van a protokollnak, amellyel növelik hatékonyságot pl. a Wi-Fi hálózatok esetén? [**8 pont**]

- CSMA = Vivőérzékeléses többszörös hozzáférés, CA: Collision Avoidance kiegészítés!
- Akkor van rá szükség, ha a küldő nem tudja biztosan érzékelni, hogy ütközés miatt nem érkezett meg az elküldött csomagja. A fentiek miatt szükséges az ütközés detektálás helyett inkább az ütközés elkerülésére helyezni a hangsúlyt. [**1 pont**]
- Adás előtt egy csomópont belehallgat a csatornába. [**1 pont**] Ha üres, akkor sem azonnal, hanem csak egy véletlen idő múlva küldi a jelet. [**1 pont**]
- Az ütközést nem figyeli, így a csomópont, ha elkezdte adni a keretet, akkor azt teljesen el is küldi. [**1 pont**]
- Nyugtázás: az ütközésérzékelés hiányában a vevőtől nyugtát vár a feladó, hogy biztosan tudja, hogy megérkezett az adat. A feladó a nyugtára egy időzítő segítéségével várakozik. Ha nem érkezik meg a nyugta időben, akkor újraküldi a keretet. [**2 pont**]
- A küldő egy adáskérés (Request to Send – RTS) üzenetet küld, mielőtt továbbítaná a csomagját. Ezzel kér engedélyt és időrést a keret továbbítására. Az RTS tartalmazza a keret továbbításához szükséges időt. Természetesen az RTS is ütközhet más jelekkel, de ez egy kisméretű keret, így ütközés esetén kisebb a veszteség. A hozzáférési pont egy adásra kész (Clear to Send – CTS) választ küld adatszórással, amennyiben elfogadja a kérést. Az adatszórásos CTS üzenetet minden csomópont megkapja, és az abban jelzett ideig nem fogják zavarni a kommunikációt. [**2 pont**]

6. Feladat Adott az alább látható 16 bites adat. Alkalmazzon rajta 2 dimenziós paritásellenőrzést: számítsa ki és írja le a szükséges paritásbiteket! Magyarázza el, hogyan kell képezni a paritásbiteket! Tegyük fel, hogy az átvitel során a 7. bit elromlik (1 helyett, 0 érkezik meg). Mutassa meg és magyarázza el, hogyan tudja ezt a hibát kezelni a 2 dimenziós paritásellenőrzés! [**6 pont**]

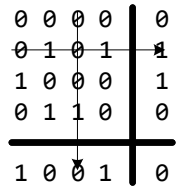
0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 0

- A **kétdimenziós paritásellenőrzés** esetén az adatbiteket egy mátrixba rendezzük és minden egyes sorhoz, illetve oszlophoz paritásbiteket számítunk. [**1 pont**]
- Akkor 0 a paritásbit, ha az adatban lévő 1-es bitek száma páros, egyébként páratlan. [**1 pont**]
- Fontos, hogy a mátrix jobb alsó sarkában az összes bit együttes paritásbitjét számítjuk ki! [**1 pont**]

0	0	0	0	0
0	1	1	1	1
1	0	0	0	1
0	1	1	0	0
1	0	0	1	0

[**1 pont**]

Ha egy bit romlott el, akkor abban a sorban, illetve oszlopban helytelen lesz a paritásbit. A hibás sor és oszlop egyértelműen kijelöli a hibás cellát! [**2 pont**]



7. Feladat Készítsen HTML lapot, amely kiírja a $_SESSION asszociatív tömb valamennyi kulcs-érték párosát táblázatos formában! (A példában feltételezhető, hogy az asszociatív tömb elemeinek típusa skalár.) [**4 pont**]

```
<?php
    session_start();                                  // 1 pont
?>
<htlml>
<body>
    <h1> PHP List All Session Variables</h1>
    <table>
<?php
    foreach ($_SESSION as $key=>$val)                          // 2 pont
      echo "<tr><td>".$key. "</td><td>".$val."</td></tr>";     // 1 pont
?>
</table>
</body>
</html>
```

8. Feladat Ismertesse a Chomsky féle nyelvosztályokat és főbb jellemzőit, valamint adjon példát az egyes nyelvosztályba tartozó nyelvekre. [**8 pont**]

Chomsky a helyettesítési szabályok komplexitása alapján négy nyelvosztályt definiált, s ezeket számokkal jelölte 0-tól 3-ig.

3-as nyelvosztály - **Reguláris nyelvek**

A reguláris nyelvtanban csak az alábbi szabálytípusok alkalmazhatók:
Jobbreguláris nyelvtan esetén: $A \to a \mid bB$
Balreguláris nyelvtan esetén pedig: $A \to a \mid Bb$
A reguláris nyelvek megengedik az $A \to \varepsilon$ szabály használatát is, ahol ε az üres szöveg.

Ebbe a nyelvosztályba tartoznak a programozás építőköveinek leírásai, például: a változók, számok, kulcsszavak feldolgozása. Minden véges számú mondatot tartalmazó nyelv a reguláris nyelvek családjába tartozik. Néhány konkrét példa reguláris nyelvekre: $a^i \quad i > 0$, $a^i b^j \quad i, j > 0$.

A reguláris nyelvek véges automatával feldolgozhatóak. A reguláris nyelvek mondatainak levezetései gereblye jellegű bináris fák.

2-es nyelvosztály - **Környezetfüggetlen nyelvek** (CFG - context free grammar)

A környezetfüggetlen nyelvek osztályába tartozó nyelvek levezetési szabályai: A $\to \alpha$ formájúak. Bal oldalon egy nemterminális áll, míg a jobb oldalon tetszőleges (terminálisok és nemterminálisok sorozata). Környezetfüggetlen nyelvek esetén is megengedett az $A \to \varepsilon$ szabály használatára.

A programozási nyelveknél például az aritmetikai kifejezések leírása, valamint minden olyan struktúrának a megadása, amely egymásba ágyazható szerkezeteket (nyitó és csukó zárójelek, begin és end, …) tartalmaz, ezzel a nyelvosztállyal adható meg. Néhány konkrét példa CF nyelvekre: $a^i b^i \quad i > 0$. A palindrom (tükör jellegű mondatokat tartalmazó mondatokat) nyelvek is CF nyelvek: pl. $S=ww^{-1}$, ahol $w=(a+b)^+$.

A CF nyelvek nem determinisztikus veremautomatával dolgozhatók fel. A CF nyelv mondataihoz tartozó levezetési gráf fa jellegű.

1-es nyelvosztály - **Környezetfüggő nyelvek** (CSG - context sensitive grammar)

A környezetfüggő nyelvek helyettesítési szabályainak korlátait kétféle módon is jellemezhetjük.
Az első megadási mód szerint a levezetési szabályok alakja: $\beta A\gamma \rightarrow \beta\alpha\gamma$ Ez a szabály azt jelenti, hogy egy nemterminális csak akkor helyettesíthető a megfelelő terminálisok és nemterminálisok sorozatával, ha előtte és utána a szabály által előírt terminálisok és nemterminálisok állnak.
Második megadási mód szerint: $\alpha \rightarrow \beta$, ahol $|\alpha| \leq |\beta|$
vagyis a helyettesítési szabály jobb oldalának hossza nem lehet rövidebb a bal oldal hosszánál. Ezen tulajdonság alapján ezeket a nyelveket nem csökkentő nyelveknek is nevezik.

A $a^i b^i c^i \quad i > 0$ mondatokat tartalmazó nyelv CS nyelv. A másoló jellegű nyelv (copy language) is CS nyelv. Pl: $S=ww$, ahol $w=(a+b)^+$.

A CF nyelv korlátos hosszú Turing géppel dolgozható fel. A CF nyelv mondatainak levezetése gráf jellegű. Kereszt jellegű függőségeket is tartalmazhat.

0-ás nyelvosztály – rekurzíven felsorolható nyelvek

A 0-ás nyelvosztályban alkalmazható helyettesítési szabályokra nincs megkötés. A levezetési szabályok általános formája: $\alpha \rightarrow \beta$, ahol semmilyen megkötés nincsen α és β hosszára. Ezeket a nyelveket rekurzíven felsorolható nyelveknek is nevezik.

Annak ellenére, hogy az $A \rightarrow \varepsilon$ szabály formailag a 0-s nyelvosztályba tartozik, megengedjük megjelenését valamennyi nyelvosztályban.

A rekurzívan feldolgozható nyelvek Turing géppel dolgozhatók fel.

9. Feladat Készítsen determinisztikus automatát az alábbi szabályrendszerrel leírható szövegek elemzéséhez:

$$S \rightarrow aSS|bSS|c$$

Elemezze az $abccc$ szöveget az elkészített automata segítségével!

A fenti nyelvtan LL(1) elemezhető, mivel az alternatívák közül egy karakter előre olvasásával választani tudunk. A veremautomata elemző táblázata: [**5 pont**]

	a	b	c	#
S	aSS	bSS	c	
a	pop			
b		pop		
c			pop	
#				o.k.

Az abccc szöveg elemzése: [**4 pont**]

Bemenet	Verem
abccc#	S#

abccc#	aSS#
bccc#	SS#
bccc#	bSSS#
ccc#	SSS#
ccc#	cSS#
cc#	SS#
cc#	cS#
c#	S#
c#	c#
#	# - elfogadva