Rajzolja fel a digitális számítógép Neumann-féle modelljének ***blokkvázlatát***, ***sorolja*** fel a modell működését meghatározó ***alapelveket!***

Mi ***különbözteti*** meg egymástól a memóriában tárolt ***utasításokat és adatokat*** egymástól?

Sorolja fel milyen tényezőktől függ egy számítógép teljesítménye!

Magyarázza el mi az előnye, illetve a hátránya az általános, illetve a speciális célú regiszter-használatnak!

Sorolja fel, és néhány szóval jellemezze az utasításrendszer tervezési szempontjait!

Írja be, hogy ***négy-címes számítógép*** utasításai esetén mit tartalmaznak az egyes ***mezők!***

OP. Kód	operandus 1.	op 2.	eredmény	köv. utasítás

Magyarázza el, hogyan lehetett ebből ***2 címes*** megoldást létrehozni!
Adja meg milyen új ***utasítástípust***, illetve milyen speciális célú ***regisztert*** alkalmaztak, hogy ebből 2 ***címes*** megoldást hozzanak létre
Utasítástípus: vezérlésátadó ut.
Regiszter...IP v. PC... ***Feladata:***...

Magyarázza el, hogyan lehet a négycímes utasításkészletű számítógépeknél alkalmazott megoldásból 3, 2, 1, 1.5 címes megoldást kialakítani!

Ismertesse mit jelent a többkomponensű címzés, adjon egy lehetséges példát!

Milyen többkomponensű címzési mód használható előnyösen egy tömb elemeinek az elérésére, és ez hogyan állítja elő az effektív címet? indexelt
Címzési mód:... ***Effektív cím=***...

Mi a stack frame (verem keret) alkalmazásának előnye?

Miben és miért különbözik egy Pascal, illetve egy C programnyelv stack frame implementációja?

Milyen többkomponensű címzési módot alkalmaznak a stack frame esetén, mi ennek az előnye?

Ismertesse a CISC, illetve a RISC utasításkészlet jellemzőit!

Jellemezze néhány szóval az alábbi elven kialakított utasításrendszereket

CISC		RISC
sokféle, összetett	***utasítás***	kevés, egyszerű
bonyolult	***Címzési mód***	

Ismertesse milyen módszereket alkalmaznak a számítógép teljesítményének növelésére!

Sorolja fel az utasítás végrehajtás gyorsításának módszereit!

Ismertesse a RISC processzoroknál alkalmazott elveket!

Ismertesse a processzoroknál alkalmazott PIPE LINE elvét!

Milyen utasítás egymásra hatási problémák léphetnek fel a PIPE LINE alkalmazásakor?

Mi a különbség a lappangási idő, az újraindítási idő, illetve az utasítás-áteresztő képesség között?

Egy Pipe-line-t alkalmazó processzor ***három elemi*** műveletvégzőt tartalmaz. ***Az*** első elemi műveletvégző végrehajtási ideje ***50ns,*** a második ***30ns*** a harmadik ***50ns*** a részeredmény áttöltéshez szükséges időt elhanyagoljuk.

Hány ns alatt hajtódna végre ***egy*** utasítás pipe line nélkül? T_{ut}=...130...ns

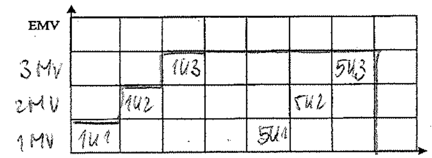
Hány ns alatt hajtódik végre ***három utasítás*** a Pipe-line működésekor? T=3·50+2·50=250...ns

Mekkora az újraindítási idő a fenti esetben ? ***Tui***=...50...ns

a) Rajzolja be a mellékelt ábrába ***öt utasítás pipe-line*** elven történő megvalósítását, ha feltételezzük, hogy minden utasítást ***három elemi műveletvégző*** dolgoz fel és ezek elemi művelet-végrehajtási ideje egyenlő.
b) Adjon egy lehetséges (gyakori) részművelet feladatot a három elemi műveletvégző egységnek (EMV)

1EMV:...
2EMV:...
3EMV:...

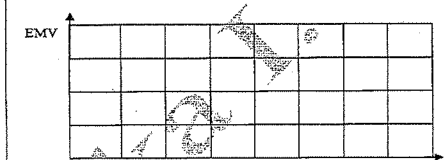
Hány ns egy utasítás végrehajtási ideje?
...ns



c) Hány ns alatt hajtódik végre az öt utasítás ha az elemi műveletvégzők végrehajtási ideje ***egyenként 40ns*** (az áttöltéshez szükséges időt elhanyagoljuk)?
5·40+2·40=280...ns

a) Rajzolja be a mellékelt ábrába ***négy utasítás pipe-line*** elven történő megvalósítását, ha feltételezzük, hogy minden utasítást ***három elemi műveletvégző*** dolgoz fel.
b) Adjon egy lehetséges (gyakori) részművelet feladatot a három elemi műveletvégző egységnek (EMV)

1EMV:...
2EMV:...
3EMV



Hány ns alatt hajtódik végre ***egy***, illetve ***négy*** utasítás, ha az első elemi műveletvégző végrehajtási ideje ***45ns,*** a második ***30ns*** a harmadik ***40ns*** a részeredmény ***áttöltéshez illetve kiadásához szükséges idő 5ns?***
Egy utasítás:...3·50=150...ns
Négy utasítás:...300...ns

A pipe-line feldolgozás egyik problémája az utasítás egymásra hatás. ***Hogyan*** oldja meg a ***i386***-os mikroprocesszor a ***procedurális*** utasítás egymásra hatást?
Hogyan oldja meg a ***486***-os mikroprocesszor a ***procedurális*** utasítás egymásra hatást?
Miben különbözik ettől a ***Pentium megoldása***?
...

Mit jelent a feldolgozási egymásrahatás, és hogyan küszöbölhető ki?

Mit jelent az adat egymásrahatás, és hogyan küszöbölhető ki?

Milyen módszerrel csökkenthető az elemi műveletvégző egységek időben egyenlőtlen terheléséből adódó probléma?

Rajzolja fel és röviden ismertesse, a processzor tehermentesítésének blokkvázlatát társprocesszorral (8086-8087), illetve DMA vezérlővel!

Magyarázza el hogyan működik időben átlapolva a processzor és a matematikai társprocesszor?

Hogyan tudja a 8086 processzor az eredményt, annak felhasználása előtt, kivárni?

Neumann-alapelvnek megfelelő számítógépekre vonatkozó alábbi kijelentések közül jelölje **x**-szel az igaz állítás(oka)t és - jellel a hamis(ak)at!

*Pontozásnál **minden jó jelölés** +0,5 pont, **minden hibás jelölés** -0,5 pont, eredő >=0*

1.	A utasítást és az adatot külön memóriában tárolja, így azok, külön sínen gyorsabb elérésűek, és a hely alapján egyértelműen azonosíthatók.	H
2.	A CPU egy –már meglévő- utasításkészlet gyorsabb implementálása (emulálása), érdekében mindig huzalozott vezérlő egységet tartalmaz.	H
3.	Az indirekt és az indexelt címzés alkalmazása előnyösen alkalmazható összetett adatszerkezetek kezelésénél.	I
4.	Négycímes utasításkészletnél nincs szükség vezérlésátadó utasításra (pl.: feltétel nélküli ugró utasításra).	I
5.	A RISC elvű processzoroknál az összetett utasítások megvalósítására gyakran mikroprogramozott vezérlőegységet alkalmaznak.	H
6.	Pipe-line alkalmazásakor az egymás után következő fokozatok (elemi műveletvégzők) között négy átmeneti tárolót kell alkalmazni. egy kell	H

7.	A be és kimenő adatokat a gyorsabb elérés érdekében az aritmetikai-logikai (ALU) egységben tárolja	H
8.	DMA vezérlő alkalmazása esetén a ki/bemenő adatok a ALU-n keresztül olvashatók be/írhatók ki a memóriába. közvetlen periféria-memória átvitel	H
9.	Multitask-os rendszereknél a fizikai és a virtuális processzor összerendelést a ko-processzor végzi. op. rendszer végzi : TASK SWITCHING	H
10.	Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat. cím → memóriahely, ahol operandus címe	H
11.	A stack frame (verem keret) alkalmazásakor a bemenő paraméterek helyének felszabadítása (keret lebontása) mindig a függvényt hívó program feladata. csak C-ben	H
12.	A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában. bemenő paraméterre használják a stack frame-t	H

13.	Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	I
14.	A utasítást és az adatot külön memóriában tárolja, az utasítást és az adatot a memóriában a tárolás formátuma különbözteti meg.	H
15.	A CISC elvű számítógépekben az utasítások nem azonos méretűek és rendszerint több óraciklus alatt hajthatók végre, és ez előnyös a pipe line alkalmazásánál. (a vége már nem igaz)	H
16.	Az ENTER és a LEAVE utasítás az x86-os processzornál a verem keret (stack frame) alkalmazását támogatja.	I
17.	A RISC processzoroknál az aritmetikai utasítások operandusai vagy regiszterben, vagy a memóriában találhatók. operandus mindig regiszterben	H
18.	A térbeli és az időbeli lokalitási elvek miatt gyorsító tárakat (cache) csak az utasítások tárolására használhatnak. külön adat-ut. / univerzális / ...	H

19.	Az utasításokat memóriában tárolja, az adatokat perifériából kapja.	H
20.	Az utasításokat és az adatokat bináris formában tárolja.	I
21.	Az utasításokat és az adatokat a memóriában csak a program algoritmusa különbözteti meg.	I
22.	Az utasításhoz és az adathoz külön-külön cím- és adatbusz tartozik. → Harvard-archit.	H

23.	A utasítást és az adatot külön memóriában tárolja, az utasítást és az adatot a memóriában a tárolás formátuma különbözteti meg.	H
24.	A RISC elvű processzoroknál a gyorsabb működés elérésére huzalozott vezérlő egységet alkalmaznak.	I
25.	RISC elvű processzoroknál a két-vagy többkomponensű címzés (pl.: bázisregiszteres és indexelt) előnyösen alkalmazható összetett adatszerkezetek kezelésénél.	H
26.	Kétcímes utasításkészletnél nincs szükség vezérlésátadó utasításra(pl.: feltétel nélküli ugró utasításra). → 2 operandus címzése + PC és vez átadó	H
27.	A CPU egy –már meglévő- utasításkészlet gyorsabb implementálása (emulálása), érdekében mikroprogramozott vezérlő egységet tartalmazhat.	I
28.	Pipe-line alkalmazásakor az egymás után következő műveletvégzők közé, a működési idő különbség miatt, átmeneti tárolót lehet alkalmazni. kell	I

29.	Az utasításokat és az adatokat az op. memóriában csak a program algoritmusa különbözteti meg.	I
30.	Az utasítást és az adatot külön memóriában tárolja, így az külön sínen gyorsabb elérésű, és a hely alapján egyértelműen azonosítható.	H
31.	Indexelt címzésnél az effektív címet az utasításban lévő címrészből és egy regiszter tartalmából állítja elő. ua.: bázisreg. címzés → teljes címtartomány	I
32.	Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	H
33.	A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben bázis-relatív címzést alkalmaznak.	I
34.	A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában.	H

35.	A utasítást és az adatot az operatív memóriában a tárolás formátuma és a helye különbözteti meg..	H
36.	A RISC elvű processzoroknál a gyorsabb működés elérésére huzalozott vezérlő egységet alkalmaznak.	I
37.	Az indirekt és az indexelt címzés alkalmazása előnyösen alkalmazható összetett adatszerkezetek kezelésénél.	I
38.	Kétcímes utasításkészletnél nincs szükség feltétel nélküli ugró utasításra, illetve vezérlésátadó utasításra.	H
39.	A CPU egy –már meglévő- utasításkészlet gyorsabb implementálása (emulálása), érdekében mikroprogramozott vezérlő egységet tartalmazhat.	I
40.	Pipe-line alkalmazásakor az egymás után következő műveletvégzők közé, a működési idő különbség miatt, átmeneti tárolót lehet alkalmazni.	I

41.	Az eredeti Neumann modellnél a kombinált BE/KI meneti egység az ALU-val nem, csak a memóriával tudott közvetlenül információt cserélni.	H
42.	A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben bázis-relatív címzést alkalmaznak.	I
43.	A RISC elvű processzoroknál a gyorsabb működés elérésére mindig mikroprogramozott vezérlő egységet alkalmaznak.	H
44.	Egycímes utasításkészletnél az utasítás címrésze a következő utasítást tartalmazó memória helyét adja meg.	H
45.	A többkomponensű címzési módok előnyösen alkalmazhatók összetett adatszerkezetek kezelésére.	I
46.	Ha az utasításkészlet tartalmaz I/O utasítást akkor annak végrehajtására mindig önálló I/O processzort kell alkalmazni.	H

→ lokális / helyi változók felsorolás.

→ indexelt: részt tartalmaz

RISC: LOAD, STORE, utasítás nem lehet mem.

Neumann: azonos formában, azonos tárolás ↔ Harvard

Az egy címes utasításkészlet csak egy operandust használhat. a marié akkumulátor regisztertsen	H
A három címes utasításkészlet egyik címe az eredmény helyét jelöli ki.	I
Kétkomponensű címzésnél az effektív címet az utasításban lévő címrészből és egy regiszter tartalmából állítja elő.	I
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	H
A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában.	H
A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben indexregiszteres többkomponensű címzést alkalmaznak.	H

Pipe-line alkalmazásakor az egymás után következő fokozatok (elemi műveletvégzők) között átmeneti tárolókat kell alkalmazni.	* I
A gyorsító tár (cache) a virtuális tár és az operatív memória közötti átvitel sebességét növeli meg. op. mem. és proc. között	H
Memória átlapolás (memory interleave, memória beékelés) esetén egy páros című bájt és a közvetlenül utána következő páratlan című ugyanabban a memóriatömbben (bank) található.	H
Egycímes utasításkészletnél az egyik operandus mindig a veremmemóriában található. operandus, eredmény is	H
A CISC elvű processzoroknál az összetett utasítások megvalósítására gyakran mikroprogramozott vezérlőegységet alkalmaznak.	I
A tárkezelő egység (memory management unit, MMU) laphibát (page fault) jelez, ha egy felhasználói módú program az operációs rendszer adataihoz próbál hozzáférni. *	H

A CPU a -már meglévő- utasításkészlet gyorsabb implementálása(emulálása), érdekében gyakran mikroprogramozott vezérlő egységet tartalmaz.	I
az utasításokat és adatokat a memóriában csak a program algoritmusa különbözteti meg.	I
Az adatok könnyű, flexibilis kezelése érdekében sokféle, bonyolult, többkomponensű címzési módokat valósíthat meg.	I
Az utasításokat és az adatokat bináris formában tárolja.	I
Legalább háromcímes utasításkészletet alkalmaz. RISC-nél tipikusan igaz	H
Az operandusok címzéséhez kevés egyszerű címzési módot alkalmaz, a memóriában található operandusokhoz csak LOAD(olvasás) és STORE(írás) típusú műveletet(címzést használ). → RISC	H

Az eredeti Neumann modellnél a kombinált BE/KI meneti egység az ALU-val nem, csak a memóriával tudott közvetlenül információt cserélni.	H
A RISC elvű processzoroknál a gyorsabb működés elérésére mikroprogramozott vezérlő egységet alkalmaznak.	H
Pipe-line alkalmazásakor, ha az egymás után következő fokozatok (elemi feldolgozó egységek) száma 2n, a teljesítmény(utasításáteresztő képesség) maximum 4n-szeresére nőhet. 2n-szeresére igaz	H
Egycímes utasításkészletnél a cím a következő utasítás helyét adja meg.	H
A többkomponensű címzési módok előnyösen alkalmazhatók összetett adatszerkezetek kezelésére.	I
Az I/O processzor, az átadott kezdőcímtől, a memóriában tárolt utasításokból álló perifériakezelő algoritmust önállóan, a CPU-val párhuzamosan hajtja végre.	I

* laphiba: hivatkozott tartalom nincs az op. memóriában

→ max. műv. végzés számlálására

Négycímes utasításkészlet esetén a program következő utasítását az éppen végrehajtott jelöli ki.	I
Kétcímes utasításkészlet esetén az eredmény mindig az akkumulátorban keletkezik.	H
Kétcímes utasításkészlet esetén mindig kell vezérlésátadó (ugró) utasítás.	I
Bázisregiszteres memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	H
A RISC processzoroknál egy ciklus alatt végrehajtható utasításokat használnak, mert ez elősegíti a pipe-line szervezést.	I
A stack frame alkalmazása esetén a szubrutinok (függvények) lokális változóinak mindig a szubrutint hívó program foglal helyet. hívott program	H

A három címes utasításkészlet egyik címe az eredmény helyét jelöli ki.	I
Az egy címes utasításkészlet csak egy operandust használhat.	H
Kétkomponensű címzésnél az effektív címet az utasításban lévő címrészből és egy regiszter tartalmából állítja elő.	I
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	H
A stack frame (verem keret) alkalmazásakor a bemenő paraméterek és a lokális változók címzésére a keretben bázis-relatív címzést alkalmaznak.	I
A stack frame a szubrutinokat (függvényeket) megvalósító algoritmusok elejét és végét jelöli ki a memóriában.	H

Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	I
A RISC elvű processzoroknál a gyorsabb működés elérésére decimális aritmetikát alkalmaznak.	H
Pipe-line esetén az utasítás egymásra hatás egyik fajtája a procedúrális egymásra hatás. Ennek elkerülésére a Pipe-line-t mindig teljesen kiürítik és újra töltik.	H
Egycímes utasításkészletnél nincs szükség vezérlésátadó utasításra (pl.:feltétel nélküli ugró utasításra).	H
A DMA egység az utasításkészletben szereplő, de a CPU-ban nem megvalósított utasításokat hajtja végre a memóriában tárolt szubrutinok felhasználásával. → CoProc	H
A többkomponensű címzési módok hátránya, hogy nem alkalmazhatók összetett adatszerkezetek kezelésére.	H

Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	* I
A RISC elvű processzoroknál a gyorsabb működés elérésére huzalozott vezérlő egységet alkalmaznak.	I
Pipe-line alkalmazásakor, ha az egymás után következő fokozatok (elemi feldolgozó egységek) száma n, a teljesítmény maximum n-szeresére nőhet.	I
Egycímes utasításkészletnél a cím az egyik operandus helyét adja meg.	I
A többkomponensű címzési módok hátránya, hogy nem alkalmazhatók összetett adatszerkezetek kezelésére.	H
A társprocesszor az utasításkészletben szereplő, de a CPU-ban nem megvalósított utasításokat (pl.:aritmetikai) önállóan, a CPU működésével párhuzamosan, hajthatja végre.	I

Pipe-line alkalmazásakor az egymás után következő két utasítás azonos típusú rész-műveleteit (két fetch, két dekódoló, stb.) azonos időszeletben egyszerre dolgozzák fel.	H
Pipe-line esetén az utasítás egymásra hatás egyik fajtája a procedúrális egymásra hatás. Ez kiküszöbölhető, ha négy utasításon belül nincs két vezérlésátadó utasítás.	H
RISC elvű processzoroknál a gyorsabb működés érdekében nem használnak mikroprogramozott vezérlő egységet.	H
A CISC elvű processzoroknál csak LOAD és STORE típusú adatmozgató utasításokat valósítanak meg.	H
A kétcímes utasításkészletnél az egyik cím az operandusok címe a másik az eredmény címe.	H
A stack frame (verem keret) a stack-ként (verem) felhasználható memóriaterület elejét és a végét jelöli ki a memóriában.	H

Négycímes utasításkészlet esetén mindig szükség van akkumulátor regiszterre.	H
Közvetlen operandusú (immediate) címzésnél az operandust az utasítást követő memóriahely tartalmazza.	I
A veremmutató (SP) tartalma x86 mikroprocesszornál aritmetikai utasításokkal módosítható, s ez felhasználható pl.: stack frame-né a lokális változó helyének lefoglalásával.	I
Az ENTER és a LEAVE utasítás a verem keret (stack frame) alkalmazását támogatja.	I
A RISC processzoroknál az aritmetikai utasítások operandusai vagy regiszterben vagy a memóriában találhatók.	H
A gyorsítótár (cache) tartalma hosszabb ideig eltérhet az operatív memória megfelelő rekeszeinek tartalmától.	H

Négycímes utasításkészlet esetén nincs szükség vezérlés átadó (pl.:ugró) utasításra.	I
Négycímes utasításkészlet esetén a program következő utasítását mindig az éppen végrehajtás alatt lévő jelöli ki.	H
A RISC processzoroknál egy ciklus alatt végrehajtható utasításokat használnak, mert ez elősegíti a pipe-line szervezést.	I
A CISC elvű processzoroknál csak LOAD és STORE típusú adatmozgató utasításokat valósítanak meg, mert ezek egyszerű címképzésűek és gyorsak.	H
Pipe-line alkalmazásakor, ha az egymás után következő fokozatok (elemi feldolgozó egységek) száma n a teljesítmény minden esetben n- szeresére nő.	H
Pipe-line esetén a feldolgozási egymásra hatás kiküszöbölhető, ha megtöbbszörözik a szükséges elemi feldolgozóegységek számát.	I

Kétcímes utasításkészlet esetén a program következő utasítását az éppen végrehajtott jelöli ki.	H
Kétcímes utasításkészlet esetén az egyik cím az eredmény helyét, míg a másik cím a következő utasítás helyét jelöli ki.	H
Pipe-line esetén a procedúrális egymásra hatás kiküszöbölhető, ha a további műveletek feldolgozását felfüggesztik a feltételes vezérlés átadás feltételének kidolgozásáig.	I
Pipe-line alkalmazásakor az egymás után következő fokozatok (elemi feldolgozó egységek) között mindig átmeneti tárolókat kell alkalmazni.	H
A RISC elvű processzoroknál csak LOAD és STORE típusú adatmozgató utasításokat valósítanak meg, mert ezek egyszerű címképzésűek és gyorsak.	I
A CISC elvű processzoroknál az összetett utasítások megvalósítására gyakran mikroprogramozott vezérlőegységet alkalmaznak.	I

Az eredeti Neumann modellnél az utasításokat és az adatokat az operatív memóriában a tárolás helye és a tárolás formátuma különbözteti meg.	H
A stack frame (verem keret) alkalmazásakor a keretet az algoritmus kódja és a hozzá tartozó adatok elválasztására használják.	H
Pipe-line alkalmazásakor az egymás után következő négy utasítás azonos típusú részműveleteit (pl.: négy fetch, négy dekód., stb.) azonos időpillanatokban dolgozzák fel.	H
Kétcímes utasításkészletnél a két cím a két operandus helyét adja meg.	H
RISC elvű processzoroknál a gyors működés és az egyszerű címzés miatt csak LOAD és STORE típusú memória-referens utasításokat valósítanak meg.	I
A CPU egy-már meglévő- utasításkészlet gyorsabb implementálása (emulálása) érdekében mikroprogramozott vezérlőegységet tartalmazhat.	I

Az eredeti Neumann modellnél a kombinált BE/KI meneti egység az ALU-val nem, csak a memóriával tudott közvetlenül információt cserélni.	H
Az utasításokat és az adatokat az operatív memóriában azonos formában tárolja, így azokat csak a program algoritmusa különbözteti meg.	H
A RISC elvű processzoroknál a gyorsabb működés elérésére mindig mikroprogramozott vezérlő egységet alkalmaznak.	H
A RISC elvű processzoroknál egyetlen ciklus alatt végrehajtható, egyforma hosszúságú utasítások kialakításával előnyösen alkalmazhatóa gyorsításra a PIPE-LINE elv.	I
Egycímes utasításkészletnél a cím a következő utasítás helyét adja meg.	H
A többkomponensű címzési módok előnyösen alkalmazhatók összetett adatszerkezetek kezelésére.	I

Az eredeti Neumann modellnél a BE -és KI meneti egység különálló volt és a memóriával nem, csak az ALU-val tudott közvetlenül információt cserélni.	I
A utasítást és az adatot fizikailag mindig külön álló memóriában tárolja, így az külön sínen gyorsabb elérésű, és a hely alapján egyértelműen azonosítható.	H
A RISC elvű processzoroknál a gyorsabb működés érdekében nem alkalmaznak mikroprogramozott vezérlő egységet.	I
A RISC elvű processzoroknál az egyszerű címzés miatt csak LOAD és STORE típusú műveletekkel érik el a memóriában lévő adatokat.	I
Kétcímes utasításkészletnél nincs szükség vezérlésátadó utasításra(pl.: feltétel nélküli ugró utasításra).	H
Indirekt memória címzésnél az utasítás címrésze a következő utasítást tartalmazó memóriahelyre mutat.	H

A ***RISC***-alapelvnek megfelelő számítógépekre vonatkozó alábbi kijelentések közül jelölje x-szel az igaz(ak)at és - jellel a hamis(ak)at!

A CPU-nál a gyorsabb működéshez szükséges egyenlő számú órajelciklust igénylő utasítások érdekében nem alkalmazzák a pipe-line elvet.	H
Az utasítást memóriában tárolja, az adatokat mikroprogram vezérlésű gyors perifériák tárolják.	H
Az adatok könnyű, flexibilis kezelése érdekében bonyolult, többkomponensű címzési módokat valósít meg.	H
Az utasításokat és az adatokat bináris formában a memóriában tárolja, ezér ezeket csak a program algoritmusa különbözteti meg.	H

Mikroprogramozott vezérlő egységet tartalmaz.	H
Az utasítást memóriában tárolja, az adatokat perifériából kapja.	H
Az utasításhoz és az adathoz külön-külön cím- és adatbusz tartozik.	H
Az utasításokat és az adatokat bináris formában tárolja.	I

A CPU a -már meglévő- utasításkészlet gyorsabb implementálása(emulálása), valamint a gyorsabb működés érdekében mikroprogramozott vezérlő egységet tartalmaz.	H
Négycímes utasításkészletet alkalmaz.	H
Az utasításokat és az adatokat bináris formában tárolja.	I
Az általában egyetlen ciklus alatt végrehajtható, egyforma hosszúságú utasítások kialakítása miatt, előnyösen alkalmazható a PIPE-LINE elv a gyorsításra.	I
az utasításokat és az adatokat a memóriában csak a program algoritmusa különbözteti meg.	H
Az operandusok címzéséhez kevés egyszerű címzési módot alkalmaz, a memóriában található operandusokhoz csak LOAD(olvasás) és STORE(írás) típusú műveletet(címzést) használ.	I

Magasszintű programozási nyelveknél függvényhívás (szubrutinhívás) implementálásakor a bemenő paraméterek átadására, illetve a lokális változók tárolására a ***verem keretet*** (stack frame) alkalmazzák. Milyen címzési módot használnak a ***bemenő paraméterek*** és a ***lokális változók elérésére***, mi a módszer előnye?

Címzési mód:......bázisregiszteres

Előnye:...fizikai memóriától és a futás környezettől független bázis + offset

Egy *Pascal* programban adott a következő ***függvény***:

function f(i,j,k:integer):integer;
var m,n,:integer;
begin ....
end;

8086-os processzornál az *f(3,1,5)* függvényhívás után ***rajzolja*** fel (írja be a memória rekeszek tartalmát) a ***verem keretet*** (stack frame), feltételezve, hogy hívás előtt az SP a jelölt helyre mutat (az integer 16 bites)! ***Jelölje*** be a felépített keret bázisát (BP). ***Jelölje*** be a ***stack pointer*** helyét a függvény végrehajtása alatt, valamint a visszatérés után!

Milyen címzési módot használnak a ***bemenő paraméterek*** és a ***lokális változók elérésére***, mi a módszer előnye?

Címzési mód:....bázisregiszteres

Előnye:...................................

Mem.cím(hexa)	
EFA02	
SS:SP →EFA00	
EF9FE	3
EF9FC	1
EF9FA	5
EF9F8	CS
EF9F6	IP
EF9F4	BP
EF9F2	m
EF9F0	n
SP →EF9EE	

Egy *Pascal* programban adott a következő ***függvény***:

function f(i;j:integer):integer;
var m:integer;
begin ....
end;

8086-os processzornál a fenti függvényt meghívtuk az aktuális paraméterekkel. A mellékelt ábrán a memóriának az a része látható amelybe a verem keret használatakor szokásos szerkezet (stack frame) is felépült (az integer 16 bites). Az SP a keret felépítése után a végrehajtás alatt a bejelölt helyre mutat. Adja meg milyen aktuális paraméter értékekkel hívták meg a függvényt.

i=...............H *j*=..............H

Adja meg a ***keret aktuális*** (fügvény végrehajtása alatti) hexa értékét.

Keret:........................H

C nyelvű függvény esetén a bemenő paraméterek helyének felszabadítása a hívó program feladata.
Miért?..

C nyelvű függvény esetén a bemenő paramétereket fordított sorrendben kell a frame-be írni.
Miért?

Mem.cím(hexa)	
EFA02	1221H
EFA00	1300H
EF9FE	0013H
EF9FC	0001H
EF9FA	IP
EF9F8	BP
EF9F6	0A00H
SS:SP →EF9F4 végrehajtás alatt	0A01H
EF9F2	1221H
EF9F0	0012H
EF9EE	0BC2H

Egy *Pascal* programban adott a következő ***függvény***:

function f(k;l:integer):integer;
var m:integer;
begin ....
end;

8086-os processzornál a fenti függvényt meghívtuk az aktuális paraméterekkel. A mellékelt ábrán a memóriának az a része látható amelybe a verem keret használatakor szokásos szerkezet (stack frame) is felépült (az integer 16 bites). Az SP a keret felépítése után a végrehajtás alatt a bejelölt helyre mutat. Adja meg,hogy a ***k*** paraméter milyen aktuális értékével hívták meg a függvényt.

k=..............H

Mi az ***m*** lokális változó pillanatnyi értéke *m*=..............H

Mi a stackpointer értéke közvetlenül a visszatérés után?

SS:SP=...................H

Milyen címzési módot használnak a ***bemenő paraméterek*** és a ***lokális változók elérésére***, mi a módszer előnye?

Címzési mód:..

Előnye:...

Mem.cím(hexa)	
EFA02	1221H
EFA00	1500H
EF9FE	0027H
EF9FC	0002H
EF9FA	IP
EF9F8	BP
EF9F6	2A00H
SS:SP →EF9F4 végrehajtás alatt	0A01H
EF9F2	1221H
EF9F0	0012H
EF9EE	0BC2H

Ismertesse a memóriák hierarchikus felépítését!

Mit jelent a térbeli, illetve az időbeli lokalitási elv? Adjon egy-egy példát, ami alátámasztja, illetve ami sérti az előbbi elveket!

Ismertesse a lokalitási elvet , mit jelent a ***találati arány*** (Hit rate)?

Mi a cache szerepe?

Rajzolja fel egy direkt szervezésű cache ***blokkvázlatát***!