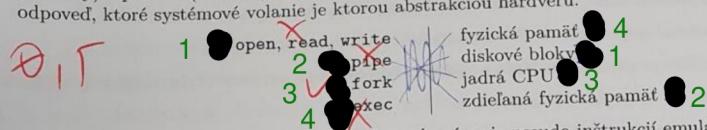


1. (2 body) Správne (čiarami) spojte výrazy z ľavého stĺpca s výrazmi v pravom stĺpco. Hľadajte odpoved, ktoré systémové volanie je ktorou abstrakciou hardvéru.



2. (2 body) Pomocou číslования od 1 zoradte vykonávanie pseudo inštrukcií emulátora:

- read next instruction 2
- for True: 1
- execute instruction (updating CPU state) 4
- decode instruction 3

3. (1 bod) Zvolte správne tvrdenie ohľadom xv6 jadra:

- je monolitické  
 je mikrojadrom  
 ide o hybridný model medzi mikrojadrom a monolitickým jadrom  
 ide o jadro OS reálneho času

4. (1 bod) Zvolte správne tvrdenie ohľadom spustenia xv6 v ladiacom režime:

- make xv6-qemu  
 make xv6-gdb  
 make gdb  
→  make qemu-gdb

5. (1 bod) Kooperatívne plánovanie procesov známená (vyberte jednu možnosť), že:

- proces sa musí sám rozhodnúť vzdaf sa procesora  
 sa procesu odoberie procesor v závislosti od jeho priority  
 proces na základe dočasného zvýšenia priority dokáže prevziať procesor  
 sa procesu odoberie procesor na základe prerušenia časovača

6. (1 bod) Vyberte pravdivé tvrdenie:

- užívateľský program dokáže dereferencovať aj virtuálne, aj fyzické adresy  
 užívateľský program dokáže dereferencovať buď virtuálne, alebo fyzické adresy, no nikdy nie oboje súčasne  
→  užívateľský program nikdy nedokáže dereferencovať virtuálne adresy  
→  užívateľský program nikdy nedokáže dereferencovať fyzické adresy

7. (1 bod) Čo sa stane pri dereferencovaní smerníka NULL v kóde používateľského procesu xv6? Vyberte optimálne tvrdenie:

- takáto dereferencia nie je možná  
→  dereferencia bude úspešná  
 vygeneruje sa výnimka výpadku stránky  
 ide o nedefinovaný stav, následkom ktorého sa vygeneruje generálna výnimka ochrany

8. (1 bod) Izolácia procesov v kontexte prednášok znamená (vyberte najvhodnejšie tvrdenie):

- vynútenú integráciu z dôvodu obnovy dôsledkov zlyhaní
- dobrovoľnú separáciu z dôvodu zapúzdrenia dôsledkov zlyhaní
- dobrovoľnú integráciu z dôvodu obnovy dôsledkov zlyhaní
- vynútenú separáciu z dôvodu zapúzdrenia dôsledkov zlyhaní

9. (1 bod) Medzi hlavné nástroje izolácie používateľského procesu patrí (vyberte správne tvrdenie):

- fyzický adresný priestor procesu
- fyzický adresný priestor jadra
- virtuálny adresný priestor procesu
- virtuálny adresný priestor jadra

10. (1 bod) Procesor RISC-V má (vyberte správne tvrdenie):

- 2 módy činnosti: user a supervisor
- 3 módy činnosti: user, machine a supervisor
- 4 módy činnosti: user, machine, hypervisor a supervisor

11. (1 bod) Režim supervisor procesora RISV-V v xv6 umožňuje oproti režimu user navyše (vyberte správne tvrdenie):

- pristupovať k adresám, pre ktoré nie je platné mapovanie v tabuľke stránok
- pristupovať k VIRTIO zariadeniu
- zakázať generovanie výnimky výpadku stránky
- zakázať generovanie výnimiek akéhokoľvek typu, pokým sa vykonáva kód jadra

12. (1 bod) Návratová hodnota systémového volania xv6 sa vracia používateľskému procesu pomocou (vyberte správne tvrdenie):

- špeciálneho registra procesora RISC-V
- rovnakého registra, ktorý sa používa na návratovú hodnotu funkcie
- toho registra, ktorý sa používa na prechod z priestoru používateľa do priestoru jadra
- ľubovoľného registra, ktorý sa definuje v hlavičkovom súbore `kernel/riscv.h`
- všeobecne dohodnutého registra a1

13. (1 bod) Ako reaguje nemodifikovaná verzia xv6 na výpadok stránky v režime používateľa (vyberte optimálne tvrdenie):

- ak ide o legítimnu žiadosť procesu o prístup ku pamäti, **doinštaluje** sa chýbajúce mapovanie a **reštartuje** sa kód, ktorý výpadok spôsobil
- doinštaluje sa chýbajúce mapovanie a **reštartuje** sa kód, ktorý výpadok spôsobil
- ukončením behu xv6
- ukončením procesu používateľa, ktorý výpadok vyvolal

14. (1 bod) Vlákno je: (označte správnu možnosť)

- závislé sériové vykonávanie kódu
- závislé paralelné vykonávanie kódu
- nezávislé sériové vykonávanie kódu
- nezávislé paralelné vykonávanie kódu

enie).

15. (1 bod) Označte 2 správne odpovede ohľadom implementácie vláken nemodifikovaného xv6.

vlákna jadra zdieľajú spoločnú virtuálnu pamäť  
 vlákna jadra nezdieľajú spoločnú virtuálnu pamäť  
 vlákna používateľských procesov zdieľajú spoločnú virtuálnu pamäť  
 vlákna používateľských procesov nezdieľajú spoločnú virtuálnu pamäť

16. (1 bod) Akú stratégiu plánovania procesov používa jadro xv6?

FCFS (First Come First Served)  
 SJF (Shortest Job First)  
 RR (Round Robin)  
 PS (Priority Scheduling)

17. (1 bod) Označte optimálne tvrdenie.

Zámky vždy znižujú efektivitu paralelného vykonávania.  
 Zámky vždy zvyšujú efektivitu paralelného vykonávania.  
 Zámky veľmi zriedkavo znižujú efektivitu paralelného vykonávania.  
 Zámky veľmi zriedkavo zvyšujú efektivitu paralelného vykonávania.

18. (1 bod) Najmenšia adresovateľná jednotka pri práci s diskom je: (zvolte pravdivú odpoved)

1 bajt  
 1 blok  
 1 rámc  
 1 sektor  
 1 stránka

19. (2 body) Majme i-uzol, ktorý obsahuje 3 priame, 2 nepriame a 1 dvojito nepriamy blok. Veľkosť bloku je 6 bajtov a číslo bloku zaberá 1 bajt. Zodpovedajte nasledovné otázky:

- Maximálny počet využitých nepriamych blokov je: 90
- Maximálny počet využitých priamych blokov je: 18
- Maximálna veľkosť súboru je: (3+6)+2+(6) + 6<sup>3</sup> **306 bytes**

20. (1 bod) Aké číslo i-uzla zodpovedá koreňovému priečinku? Zvoľte najlepšiu možnosť.

0  
 1

toto číslo špecifikuje návrhár súborového systému a tvorí statickú súčasť implementácie  
 toto číslo sa zistuje dynamicky pri zavádzaní súborového systému zo *super bloku*

21. (1 bod) Čo je prvotným cieľom žurnálu súborového systému? Zvoľte najšpecifickejšiu odpoved.

integrita súborového systému  
 spokojnosť používateľa  
 bezpečnosť dát  
 efektivita využitia hardvéru

22. (1 bod) Očíslovaním (od 1) označte poradie vykonávanie operácií logovacieho systému v xv6:

- 1. zápis obsahu blokov na ich cieľové miesta na disku
- 2. zápis hlavičky logu na disk
- 3. zápis N=0 do hlavičky logu na disku
- 4. synchronne čakanie na dokončenie zápisov
- 5. zapísanie blokov z *cache* do logovacej oblasti na disku

23. (1 bod) Nenulová hodnota  $N$  v hlavičke logu na disku znamená (zvolte správnu odpoved):

- počet neúspešne inštalovaných blokov logu
- počet úspešne inštalovaných blokov logu
- počet blokov logu čakajúcich na inštaláciu
- počet pokusov o obnovu súborového systému z logu

24. (1 bod) Z pohľadu kódu vykonávaného na CPU sú prerušenia (zvolte správnu odpoved):

- symetrické
- asymetrické
- synchronne
- asynchronne

25. (1 bod) CPU pri vykonávaní kódu jadra xv6 (zvolte správnu odpoved):

- nesmie mať vypnuté prerušenia
- musí mať vypnuté prerušenia
- môže mať vypnuté prerušenia
- sice môže, ale nemá vypnuté prerušenia

26. (1 bod) Kód obsluhy prerušenia v jadre xv6 (zvolte správnu odpoved):

- nesmie bežať v kontexte používateľského procesu
- musí bežať v kontexte používateľského procesu
- môže bežať v kontexte používateľského procesu
- sice môže, ale nebeží v kontexte používateľského procesu

27. (1 bod) Zvolte pravdivé tvrdenie:

- výrobcovia hardvéru rešpektujú dohodu, že IRQ číslo zariadenia je na každej platforme rovnaké
- výrobcovia hardvéru rešpektujú dohodu, že IRQ číslo *UART* zariadenia je na každej platorme rovnaké
- IRQ číslo reprezentuje chybový kód zariadenia
- IRQ číslo reprezentuje zdroj prerušenia

28. (3 body) V jazyku C uvedte príklad použitia systémového volania OS typu UNIX. Vysvetlite:

1. význam vami uvedeného SV,
2. význam jednotlivých argumentov, ak ich SV má,
3. interpretáciu návratovej hodnoty.

3

pid = fork();  $\rightarrow$  Vykonanie deťúho procesu  
do premennej pid sa uloží buď 0 - rodičovský proces,  
alebo ~~1 - deťú~~ proces. Výsledok je vykonanie  
deťúho procesu.  
 $\sim \sim$ .

#### Moje alternatívne riešenie:

a) Príkladom systémového volania OS typu UNIX môže by systemové volanie `chmod()`, ktoré slúži na zmenu prístupových práv k súboru alebo adresáru.

b) `chmod(const char \*path, mode\_t mode)`

1. path - cesta k súboru alebo adresáru, ktorého prístupové práva sa majú zmeni
2. mode - nové prístupové práva, ktoré sa majú nastavi pre daný súbor alebo adresár.

c) Systémové volanie `chmod()` vracia 0, ak sa prístupové práva úspešne zmenili, inak vracia -1 a nastavuje sa erro na hodnotu odpovedajúcu chybe.

29. (3 body) Nakreslite schému 1-úrovňového prekladu virtuálnej adresy na fyzickú.

☒

Nie som schopný kresli obrázky, ale môžem vám opísť proces 1-úrovňového prekladu virtualnej adresy na fyzickú:

1. Procesor posielá požiadavku na prístup k virtuálnej adrese.
2. Tabučka stránok (page table) sa používa na vyhadanie mapovania medzi virtuálnou adresou a fyzickou adresou.
3. Ak sa nenájde platné mapovanie, generuje sa výnimka výpadku stránky.
4. Ak sa nájde platné mapovanie, virtuálna adresa sa prekladá na fyzickú adresu pomocou offsetu a indexu do tabučky stránok.
5. Procesor môže pristupova k fyzickej adrese a vykona požadovanú operáciu.

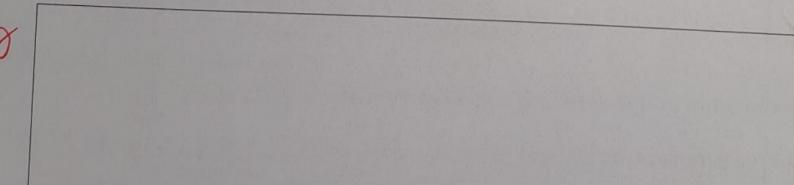
30. (3 body) V niekoľkých krococh schematicky opíšte kroky, ktoré musí robiť obsluha výpadku stránky COW forku.

- D
1. Procesor generuje výnimku výpadku stránky, ke používateský proces pristupuje k neexistujúcej stránke v virtuálnom adresnom priestore.
  2. Jadro operácie systému vykonáva rutinu obsluhy výpadku stránky, ktorá sa zvyajne volá page fault handler.
  3. Rutina obsluhy výpadku stránky zistí, i sa jedná o prípad COW forku.
  4. Ak áno, rutina vytvorí kópiu stránky do novej stránky v fyzickom adresnom priestore pre nový proces.
  5. Nastaví sa bit kópia (copy-on-write) na novej stránke, aby sa vedelo, že má by kópia stránky k dispozícii pre prípad, ke sa proces bude snaži modifikova obsah stránky.
  6. Mapuje sa nová stránka do virtuálneho adresného priestoru procesu, ktorý spôsobil výpadok stránky.
  7. Proces pokrajuje v normálnom vykonávaní a môže pristupova k obsahu stránky cez novo mapovanú fyzickú adresu.

31. (1 bod) Prečo pri COW technike v xv6 potrebujeme počítadlo referencí rámcov fyzickej pamäte?

Aby sme vedeli, čiž ho máme volniť.  
(ten rámcu)

32. (2 body) Nakreslite zjednodušený model prechodu jedného používateľského procesu na iný v xv6 z pohľadu vlákien (nie funkcií!).



Nie som schopný kresli, ale môžem vysvetli prechod jedného používateského procesu na iný v xv6 z pohadu vlákien:

1. Proces vykonáva systémové volanie prerušenia, napríklad "sleep()" alebo "wait()", ktoré signalizuje jadru, že sa chce vzda procesoru.
2. Jadro zaznamená požiadavku procesu a zavolá rutinu prerušenia pre plánovanie procesov.
3. Plánova vyberie najvyššie priorotné vlákno, ktoré je pripravené pracova a nastaví ho ako aktuálne vlákno.
4. Jadro ukoní proces vlákna, ktoré sa vzdalo procesoru, a zapisuje ho do zoznamu spiace vlákna.
5. Jadro spustí proces vlákna, ktoré bolo vybrané plánovaom.
6. Proces sa opäťovne zapisuje do zoznamu aktívnych vlákien a zaína pracova.
7. Proces sa opäťovne môže vzda procesoru a proces sa opakuje.

D

33. (2 body) Objasnite sémantiku metódy release ADT Lock.

Ako je nejaky proces zbereany na uramneny a  
jej jeho procesom ho nie ide uramniť. Tak  
AK  
Unvolim ramok, ak uva číre, ak nejaky  
vezemá myjaky proces čari na ramok tak po  
release ho sleeplock myje a može ho uva prosobi.

34. (2 body) Napíšte pseudokód metódy acquire ADT SleepLock. Interný stav zámku reprezentujú atribúty locked a FIFO fronta čakajúcich procesov queue. Metóda má k dispozícii odkaz na aktuálny proces, ktorý ju vyzval: current. Zmenu stavu procesu robte pomocou metódy setState. Preplánovanie vyvolajte pomocou funkcie yield().

while (current == locked) {  
 current = relState();  
 yield();

35. (1 bod) Prečo obsahuje i-uzol na disku informáciu o počte odkazov na súborový objekt?

Aby sme ho vedeli mať

33. ÚLOHA: Metóda release ADT Lock slúži na uvonenie zamku, ktorý bol predtým získaný pomocou metódy acquire. Po uvonení zamku sa daný objekt stáva dostupný pre iné vlákna, ktoré môžu získa zámok a pracova s objektom. Táto metóda tiež zvyjne zahruje implementáciu nejakej formy mechanizmu pre riadenie prioritou, aby sa zabezpeilo, že vlákna s vyššou prioritou majú prednos pri získavaní zamku.

34. Pozri screenshot v prieinku s názvom: pseudokod\_metody\_acquire\_ADT\_SleepLock

35. Informácia o pote odkazov na súborový objekt (tzv. link count) sa používa na správu súborových systémov, ktoré podporujú hard linky. Hard linky umožňujú vytvori viacero odkazov na rovnaký súborový objekt na disku, priom každý odkaz môže by viditeľny ako samostatný súbor. Poet odkazov na súborový objekt udáva, koko hard linkov existuje na daný súbor. Ak súbor zmažeme, tak sa zmaže až vtedy, ke poet odkazov klesne na 0.

36. (2 body) Nakreslite štruktúru logu súborového systému xv6 na disku v situácii, kedy je potvrdená disková transakcia, ale inštalačia blokov ešte nenastala alebo neskončila. Nákres vhodne a čitateľne komentujte.

I'm sorry, I am not able to create images. However, I can describe the structure of the file system log in xv6 in case of a confirmed disk transaction that has not yet been installed or completed.

In xv6, the file system log is implemented as a circular buffer of disk blocks. Each block in the buffer contains a set of disk operations (such as write, delete, etc.) that need to be performed on the file system. When a disk transaction is confirmed, the corresponding operations are added to the buffer.

In case the installation of the blocks has not yet occurred or not completed, the operations will still be present in the log buffer, but they will not yet have been applied to the actual file system structures on disk. The log buffer operates as a safety net, allowing the file system to recover in case of a crash before the operations were applied.

In the event of a crash, the file system will read through the log buffer and replay any operations that were not yet applied to the file system structures. This process is known as recovery. Once the recovery process is complete, the file system will be in a consistent state, and the operations that were present in the log buffer will have been applied to the file system structures on disk.

37. (1 bod) Prečo sa funkcie copyin() a copyout() nedajú využiť v obsluhe prerušení xv6?

**8**  
Funkcie copyin() a copyout() slúžia na kopírovanie údajov medzi pamäou jadra a pamäou používateského procesu. Tieto funkcie však vyžadujú platný kontext používateského procesu, ktorý nie je dostupný v ase obsluhy prerušenia. Obsluha prerušenia sa vykonáva v kontexte jadra, preto nie je možné použiť tieto funkcie. Tiež preto, že prerušenie môže byť vyvolané odkiacovkou, tak sa nemôže závisieť od kontextu používateského procesu.

- 8**  
38. (2 body) BONUS: Indiánske kmene každú zbabelosť trestajú veľmi prísnne. Jeden z mladých bojovníkov kmene Kiovov dostal možnosť odčiniť ju tak, že mal doniesť zo stredu tábora Apačov bojový šíp. Ale mladého Kiowa chytila už prvá apučská stráž. Slúbili mu však, že ho pustia, ak im pri bojový šíp. Ale tiež bude mať a ešte pol šípa. Ale zlomil nesmie ani jeden. Mladý návratne dá polovicu zo šípov, ktoré bude mať a ešte pol šípa. Ale zlomil nesmie ani jeden. Bojovník neboli veľmi šikovní, lebo ho chytila i druhá a tretia stráž. Dali mu také isté podmienky ako prvá stráž. Stráže si boli isté, že ich podmienku nemožno splniť. No čo Kiovovi chýbalo na obratnosti, o to bol mňudrejší. Podmienky všetkých troch stráží splnil a jeden šíp mu ešte ostal. Kolko šípov zobrať? Uvedte riešenie.

Mladý Kiovov zobrať 3 šípy, pretože každá stráž požadovala polovicu šípov, ktoré mal. Polovica z 3 šípov je 1.5 šípa, takže 3 šípy sa rozdelia na 3/2 = 1.5 šípa pre každú stráž. Keže šípy sa nedajú rozdeli na polovice, tak pre každú stráž zobere jeden šíp. Celkovo zobere 3 šípy.

Mladý bojovník zobrať 3 šípy.

Prvá stráž mu ponúkla, že ho pustí, ak im dá polovicu zo šípov,

ktoré bude mať a ešte pol šípa.

Druhá stráž mu dala tiež rovnakú podmienku.

Tretia stráž mu tiež dala rovnakú podmienku.

Ke sa vrátil do svojho kmea, zostal mu jeden šíp.

Aby sme to vyriešili matematicky:

Kiow si zobrať 1 šíp od prvej stráže

Kiow si zobrať 1/2 šípa od prvej stráže

Kiow si zobrať 1 šíp od druhej stráže

Kiow si zobrať 1/2 šípa od druhej stráže

Kiow si zobrať 1 šíp od tretiej stráže

Kiow si zobrať 1/2 šípa od tretiej stráže

Kiow si zobrať celkovo  $1 + 1/2 + 1 + 1/2 + 1 + 1/2 = 3$  šípy

Kiovovi bolo ponúknuté, že z každého šípa, ktorý donesie, musí darovať polovicu jednej stráži a ešte pol šípa. Iže z každého šípa mu zostane  $1/2 + 1/2 = 1$  celý šíp. Preto aby získal celkové číslo šípov, ktoré potrebuje pre splnenie podmienok, musí doniesť 2 šípy.