+1/1/60+

Ukázka zkouškové písemky OSY

Jméno a příjmení:
.....

Odpovězte na otázky zaškrtnutím příslušného políčka. Otázky označené znakem ♣ mohou mít více než jednu správnou odpověď. U otázek se slovní odpovědí nezaškrtávejte políčka na šedém pozadí.

Otázka 1 ♣ Software šířený pod licencí GPLv2:

- ☐ není možné prodávat za peníze
- ☐ nesmí se používat s komerčními OS (např. MS Windows)
- ☐ musí být šířen společně se zdrojovými kódy
- ☐ je možné použít v proprietárním SW, ale pouze po odstranění všech hlášek ”copyright”
- ☐ *Nic z výše uvedeného není správně*

Otázka 2 ♣ Která následující tvrzení jsou pravdivá?

- ☐ Při použití mutexu může vždy nastat deadlock – vzájemné uváznutí
- ☐ Mutex lze použít pro správný přístup ke sdíleným datům
- ☐ Mutex zaručuje, že nenastane deadlock – vzájemné uváznutí
- ☐ Mutex lze použít pro synchronizaci paralelně běžících vláken
- ☐ *Nic z výše uvedeného není správně*

Otázka 3 ♣ V roce 2018 zveřejněná zranitelnost CPU zvaná Meltdown:

- ☐ způsobí, že při vykonání škodlivého kódu se procesor přehřeje a dojde k jeho zničení
- ☐ je to chyba HW a OS s tím nemůže nic dělat
- ☐ umožňuje číst data i ze stránek, ke kterým nemá uživatelský proces přístup
- ☐ je jen další mediální bublinou
- ☐ vyskytuje se pouze u procesorů od Intelu
- ☐ *Nic z výše uvedeného není správně*

Otázka 4 Mějme následující program:

```
int a[2];
pthread_mutex_t mutex[2];

void *fce1(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[0]);
        pthread_mutex_lock(&mutex[1]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[1]);
        pthread_mutex_unlock(&mutex[0]);
    }
    pthread_exit(NULL);
}

void *fce2(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[1]);
        pthread_mutex_lock(&mutex[0]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[0]);
        pthread_mutex_unlock(&mutex[1]);
    }
    pthread_exit(NULL);
}

int main()
{
    pthread_t tid[2];
    a[0]=0; a[1]=1;
    pthread_mutex_init(&mutex[0], NULL);
    pthread_mutex_init(&mutex[1], NULL);
    pthread_create(&tid[0], NULL, fce1, NULL);
    pthread_create(&tid[1], NULL, fce2, NULL);
    pthread_join(tid[0], NULL);
    pthread_join(tid[1], NULL);
    return 0;
}
```

- ☐ program vždy skončí chybou
- ☐ program někdy skončí, někdy uvázne v deadlocku
- ☐ program vždy uvázne v deadlocku
- ☐ program vždy bez problémů skončí
- ☐ program neskončí, obsahuje nekonečnou smyčku

Otázka 5 ♣ Stárnutí (starvation) je problémem, který hrozí plánovacím algoritmům. Které algoritmy ohrožuje stárnutí:

- ☐ cyklické plánování – round robin
- ☐ prioritní plánování
- ☐ First Come First Served – podle pořadí spuštění
- ☐ nejkratší proces první – shortes process next
- ☐ *Nic z výše uvedeného není správně*

Otázka 6 ♣ V tabulce stránek 32bitového systému x86 je pro virtuální adresu 0x12345678 uvedena hodnota 0xCCCCC005.

```
enum {
  PRESENT = 1<<0,
  RW      = 1<<1,
  USER    = 1<<2,
  ACCESS  = 1<<5,
  DIRTY   = 1<<6,
};
```

Které z následujících tvrzení je pravdivé:

- ☐ Jádro OS může z této stránky číst
- ☐ Uživatelský proces může do této stránky zapisovat
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC003
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC678
- ☐ Uživatelský proces může z této stránky číst
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC123
- ☐ *Nic z výše uvedeného není správně*

Otázka 7 Předpokládejte, že nedojde k chybě při spuštění následujícího programu:

```
int main() {
  int f=fork();
  f=fork();
  f=fork();
  printf("%d\n", f);
  return 0;
}
```

Jaký bude výstup tohoto programu?

- ☐ osm nenulových čísel
- ☐ šest nenulových čísel a dvě 0
- ☐ dvě nenulová čísla a dvě 0
- ☐ čtyři nenulová čísla a čtyři 0
- ☐ čtyři nenulová čísla a dvě 0
- ☐ tři nenulová čísla a jedna 0
- ☐ osm 0

Otázka 8 ♣ Hardwarově asistovaná virtualizace (např. VT-x)

- ☐ zrychluje běh virtuálního uživatelského režimu.
- ☐ odstraňuje nutnost emulovat hardware (disk, síťová karta, …) mechanismem trap-and-emulate.
- ☐ funguje jedině na 32bitovém systému.
- ☐ zrychluje vykonávání některých privilegovaných instrukcí.
- ☐ přidává další vrstvu stránkovacích tabulek.
- ☐ *Nic z výše uvedeného není správně*

Otázka 9 ♣ Jaký je vztah pojmů „page fault“ (výpadek stránky) a „segmentation fault“

- ☐ Segmentation fault běžně nastává během běhu většiny programů
- ☐ Page fault běžně nastává během běhu většiny programů
- ☐ Segmentation fault je vždy důsledek page fault
- ☐ Dva různé pojmy pro stejnou věc
- ☐ Page fault je vždy důsledek segmentation fault
- ☐ Page fault se používá k implementaci copy-on-write
- ☐ *Nic z výše uvedeného není správně*

Otázka 10 ♣ K zajištění konzistence souborového systému po náhlém vypnutí či pádu systému

- ☐ je potřeba vybavit počítač záložním zdrojem (tzv. UPS).
- ☐ je potřeba používat RAID1 a vyšší.
- ☐ je vždy potřeba kompletní kontrola souborového systému po následném zapnutí počítače.
- ☐ je potřeba provádět všechny související modifikace souborového systému atomicky.
- ☐ lze použít metodu, kdy se popis potřebných modifikací nejprve uloží do speciální oblasti na disku.
- ☐ *Nic z výše uvedeného není správně*

Otázka 11 Uvažujte následující příkaz BASHe:

```
l=`ps`
```

Který z následujících příkazů vytiskne na každou řádku jeden proces?

☐ echo ${l} ☐ echo ($l) ☐ echo ”$l” ☐ echo ’$l’ ☐ echo $l

Otázka 12 Vlákna na jedno-procesorovém počítači (bez hyper-threadingu):

- ☐ mohou běžet paralelně pokud nikdy nepoužívají sdílené proměnné
- ☐ nemohou běžet paralelně nikdy
- ☐ mohou běžet paralelně pouze pokud použijí mutex
- ☐ mohou běžet paralelně

Otázka 13 Co provádí následující příkaz? tr -d 't'

- ☐ Změní všechny znaky d na znak t
- ☐ Odstraní znaky t ze standardního vstupu
- ☐ Zkopíruje standardní vstup beze změny
- ☐ Odstraní tabulátory ze standardního vstupu

Otázka 14 Provedení funkce z jádra operačního systému na architektuře x86 lze z uživatelského programu vyvolat:

- ☐ instrukcí int s registrem obsahujícím číslo služby jádra
- ☐ instrukcí nop s registrem obsahujícím číslo služby jádra
- ☐ instrukcí call na adresu služby jádra
- ☐ instrukcí jmp na adresu služby jádra

Otázka 15 Popište, jak probíhá odesílání dat z aplikace na síť. Co se s daty děje a jakou roli v tom hraje jádro OS?

☐0 ☐0.5 ☐1 ☐1.5 ☐2

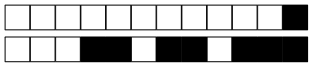
.....

.....

.....

.....

.....



+2/1/54+

Ukázka zkouškové písemky OSY

Jméno a příjmení:
.....

Odpovězte na otázky zaškrtnutím příslušného políčka. Otázky označené znakem ♣ mohou mít více než jednu správnou odpověď. U otázek se slovní odpovědí nezaškrtávejte políčka na šedém pozadí.

Otázka 1 Uvažujte následující příkaz BASHe:

```
l=`ps`
```

Který z následujících příkazů vytiskne na každou řádku jeden proces?

☐ echo "$l" ☐ echo ($l) ☐ echo $l ☐ echo '$l' ☐ echo ${l}

Otázka 2 Co provádí následující příkaz? tr -d 't'

- ☐ Odstraní tabulátory ze standardního vstupu
- ☐ Změní všechny znaky d na znak t
- ☐ Zkopíruje standardní vstup beze změny
- ☐ Odstraní znaky t ze standardního vstupu

Otázka 3 ♣ Jaký je vztah pojmů „page fault" (výpadek stránky) a „segmentation fault"

- ☐ Page fault se používá k implementaci copy-on-write
- ☐ Segmentation fault je vždy důsledek page fault
- ☐ Segmentation fault běžně nastává během běhu většiny programů
- ☐ Page fault běžně nastává během běhu většiny programů
- ☐ Dva různé pojmy pro stejnou věc
- ☐ Page fault je vždy důsledek segmentation fault
- ☐ *Nic z výše uvedeného není správně*

Otázka 4 ♣ V roce 2018 zveřejněná zranitelnost CPU zvaná Meltdown:

- ☐ vyskytuje se pouze u procesorů od Intelu
- ☐ způsobí, že při vykonání škodlivého kódu se procesor přehřeje a dojde k jeho zničení
- ☐ je to chyba HW a OS s tím nemůže nic dělat
- ☐ umožňuje číst data i ze stránek, ke kterým nemá uživatelský proces přístup
- ☐ je jen další mediální bublinou
- ☐ *Nic z výše uvedeného není správně*

Otázka 5 ♣ Stárnutí (starvation) je problémem, který hrozí plánovacím algoritmům. Které algoritmy ohrožuje stárnutí:

- ☐ nejkratší proces první – shortes process next spuštění
- ☐ prioritní plánování
- ☐ First Come First Served – podle pořadí
- ☐ cyklické plánování – round robin
- ☐ *Nic z výše uvedeného není správně*

Otázka 6 Vlákna na jedno-procesorovém počítači (bez hyper-threadingu):

- ☐ mohou běžet paralelně pokud nikdy nepoužívají sdílené proměnné
- ☐ mohou běžet paralelně pouze pokud použijí mutex
- ☐ nemohou běžet paralelně nikdy
- ☐ mohou běžet paralelně

Otázka 7 ♣ Hardwarově asistovaná virtualizace (např. VT-x)

- ☐ přidává další vrstvu stránkovacích tabulek.
- ☐ zrychluje vykonávání některých privilegovaných instrukcí.
- ☐ odstraňuje nutnost emulovat hardware (disk, síťová karta, …) mechanismem trap-and-emulate.
- ☐ funguje jedině na 32bitovém systému.
- ☐ zrychluje běh virtuálního uživatelského režimu.
- ☐ *Nic z výše uvedeného není správně*

Otázka 8 Mějme následující program:

```
int a[2];
pthread_mutex_t mutex[2];

void *fce1(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[0]);
        pthread_mutex_lock(&mutex[1]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[1]);
        pthread_mutex_unlock(&mutex[0]);
    }
    pthread_exit(NULL);
}

void *fce2(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[1]);
        pthread_mutex_lock(&mutex[0]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[0]);
        pthread_mutex_unlock(&mutex[1]);
    }
    pthread_exit(NULL);
}

int main()
{
    pthread_t tid[2];
    a[0]=0; a[1]=1;
    pthread_mutex_init(&mutex[0], NULL);
    pthread_mutex_init(&mutex[1], NULL);
    pthread_create(&tid[0], NULL, fce1, NULL);
    pthread_create(&tid[1], NULL, fce2, NULL);
    pthread_join(tid[0], NULL);
    pthread_join(tid[1], NULL);
    return 0;
}
```

- ☐ program neskončí, obsahuje nekonečnou smyčku
- ☐ program vždy uvázne v deadlocku
- ☐ program vždy skončí chybou
- ☐ program někdy skončí, někdy uvázne v deadlocku
- ☐ program vždy bez problémů skončí

Otázka 9 ♣ V tabulce stránek 32bitového systému x86 je pro virtuální adresu 0x12345678 uvedena hodnota 0xCCCCC005.

```
enum {
  PRESENT = 1<<0,
  RW      = 1<<1,
  USER    = 1<<2,
  ACCESS  = 1<<5,
  DIRTY   = 1<<6,
};
```

Které z následujících tvrzení je pravdivé:

- ☐ Jádro OS může z této stránky číst
- ☐ Uživatelský proces může do této stránky zapisovat
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC003
- ☐ Uživatelský proces může z této stránky číst
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC123
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC678
- ☐ *Nic z výše uvedeného není správně*

Otázka 10 ♣ K zajištění konzistence souborového systému po náhlém vypnutí či pádu systému

- ☐ je potřeba používat RAID1 a vyšší.
- ☐ je vždy potřeba kompletní kontrola souborového systému po následném zapnutí počítače.
- ☐ je potřeba provádět všechny související modifikace souborového systému atomicky.
- ☐ je potřeba vybavit počítač záložním zdrojem (tzv. UPS).
- ☐ lze použít metodu, kdy se popis potřebných modifikací nejprve uloží do speciální oblasti na disku.
- ☐ *Nic z výše uvedeného není správně*

Otázka 11 Popište, jak probíhá odesílání dat z aplikace na síť. Co se s daty děje a jakou roli v tom hraje jádro OS? ☐0 ☐0.5 ☐1 ☐1.5 ☐2

.....

.....

.....

.....

.....

Otázka 12 ♣ Která následující tvrzení jsou pravdivá?

- ☐ Při použití mutexu může vždy nastat deadlock – vzájemné uváznutí
- ☐ Mutex zaručuje, že nenastane deadlock – vzájemné uváznutí
- ☐ Mutex lze použít pro správný přístup ke sdíleným datům
- ☐ Mutex lze použít pro synchronizaci paralelně běžících vláken
- ☐ *Nic z výše uvedeného není správně*

Otázka 13 Provedení funkce z jádra operačního systému na architektuře x86 lze z uživatelského programu vyvolat:

- ☐ instrukcí int s registrem obsahujícím číslo služby jádra
- ☐ instrukcí jmp na adresu služby jádra
- ☐ instrukcí call na adresu služby jádra
- ☐ instrukcí nop s registrem obsahujícím číslo služby jádra

Otázka 14 ♣ Software šířený pod licencí GPLv2:

- ☐ musí být šířen společně se zdrojovými kódy
- ☐ je možné použít v proprietárním SW, ale pouze po odstranění všech hlášek ”copyright”
- ☐ nesmí se používat s komerčními OS (např. MS Windows)
- ☐ není možné prodávat za peníze
- ☐ *Nic z výše uvedeného není správně*

Otázka 15 Předpokládejte, že nedojde k chybě při spuštění následujícího programu:

```
int main() {
  int f=fork();
  f=fork();
  f=fork();
  printf("%d\n", f);
  return 0;
}
```

Jaký bude výstup tohoto programu?

- ☐ čtyři nenulová čísla a čtyři 0
- ☐ osm 0
- ☐ čtyři nenulová čísla a dvě 0
- ☐ šest nenulových čísel a dvě 0
- ☐ osm nenulových čísel
- ☐ dvě nenulová čísla a dvě 0
- ☐ tři nenulová čísla a jedna 0

Ukázka zkouškové písemky OSY

Jméno a příjmení:
.....

Odpovězte na otázky zaškrtnutím příslušného políčka. Otázky označené znakem ♣ mohou mít více než jednu správnou odpověď. U otázek se slovní odpovědí nezaškrtávejte políčka na šedém pozadí.

Otázka 1 ♣ K zajištění konzistence souborového systému po náhlém vypnutí či pádu systému

- ☐ lze použít metodu, kdy se popis potřebných modifikací nejprve uloží do speciální oblasti na disku.
- ☐ je potřeba provádět všechny související modifikace souborového systému atomicky.
- ☐ je potřeba vybavit počítač záložním zdrojem (tzv. UPS).
- ☐ je vždy potřeba kompletní kontrola souborového systému po následném zapnutí počítače.
- ☐ je potřeba používat RAID1 a vyšší.
- ☐ *Nic z výše uvedeného není správně*

Otázka 2 ♣ V tabulce stránek 32bitového systému x86 je pro virtuální adresu 0x12345678 uvedena hodnota 0xCCCCC005.

```
enum {
  PRESENT = 1<<0,
  RW      = 1<<1,
  USER    = 1<<2,
  ACCESS  = 1<<5,
  DIRTY   = 1<<6,
};
```

Které z následujících tvrzení je pravdivé:

- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC003
- ☐ Jádro OS může z této stránky číst
- ☐ Uživatelský proces může z této stránky číst
- ☐ Uživatelský proces může do této stránky zapisovat
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC123
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC678
- ☐ *Nic z výše uvedeného není správně*

Otázka 3 ♣ Hardwarově asistovaná virtualizace (např. VT-x)

- ☐ funguje jedině na 32bitovém systému.
- ☐ odstraňuje nutnost emulovat hardware (disk, síťová karta, …) mechanismem trap-and-emulate.
- ☐ zrychluje vykonávání některých privilegovaných instrukcí.
- ☐ přidává další vrstvu stránkovacích tabulek.
- ☐ zrychluje běh virtuálního uživatelského režimu.
- ☐ *Nic z výše uvedeného není správně*

Otázka 4 ♣ Software šířený pod licencí GPLv2:

- ☐ není možné prodávat za peníze
- ☐ nesmí se používat s komerčními OS (např. MS Windows)
- ☐ musí být šířen společně se zdrojovými kódy
- ☐ je možné použít v proprietárním SW, ale pouze po odstranění všech hlášek ”copyright”
- ☐ *Nic z výše uvedeného není správně*

Otázka 5 Provedení funkce z jádra operačního systému na architektuře x86 lze z uživatelského programu vyvolat:

- ☐ instrukcí call na adresu služby jádra
- ☐ instrukcí nop s registrem obsahujícím číslo služby jádra
- ☐ instrukcí int s registrem obsahujícím číslo služby jádra
- ☐ instrukcí jmp na adresu služby jádra

Otázka 6 Mějme následující program:

```
int a[2];
pthread_mutex_t mutex[2];

void *fce1(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[0]);
        pthread_mutex_lock(&mutex[1]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[1]);
        pthread_mutex_unlock(&mutex[0]);
    }
    pthread_exit(NULL);
}

void *fce2(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[1]);
        pthread_mutex_lock(&mutex[0]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[0]);
        pthread_mutex_unlock(&mutex[1]);
    }
    pthread_exit(NULL);
}

int main()
{
    pthread_t tid[2];
    a[0]=0; a[1]=1;
    pthread_mutex_init(&mutex[0], NULL);
    pthread_mutex_init(&mutex[1], NULL);
    pthread_create(&tid[0], NULL, fce1, NULL);
    pthread_create(&tid[1], NULL, fce2, NULL);
    pthread_join(tid[0], NULL);
    pthread_join(tid[1], NULL);
    return 0;
}
```

- ☐ program vždy skončí chybou
- ☐ program někdy skončí, někdy uvázne v deadlocku
- ☐ program vždy uvázne v deadlocku
- ☐ program neskončí, obsahuje nekonečnou smyčku
- ☐ program vždy bez problémů skončí

Otázka 7 Předpokládejte, že nedojde k chybě při spuštění následujícího programu:

```
int main() {
  int f=fork();
  f=fork();
  f=fork();
  printf("%d\n", f);
  return 0;
}
```

Jaký bude výstup tohoto programu?

- ☐ osm 0
- ☐ osm nenulových čísel
- ☐ tři nenulová čísla a jedna 0
- ☐ dvě nenulová čísla a dvě 0
- ☐ šest nenulových čísel a dvě 0
- ☐ čtyři nenulová čísla a čtyři 0
- ☐ čtyři nenulová čísla a dvě 0

Otázka 8 ♣ Stárnutí (starvation) je problémem, který hrozí plánovacím algoritmům. Které algoritmy ohrožuje stárnutí:

- ☐ prioritní plánování
- ☐ nejkratší proces první – shortes process next
- ☐ cyklické plánování – round robin
- ☐ First Come First Served – podle pořadí spuštění
- ☐ *Nic z výše uvedeného není správně*

Otázka 9 Popište, jak probíhá odesílání dat z aplikace na síť. Co se s daty děje a jakou roli v tom hraje jádro OS? ☐0 ☐0.5 ☐1 ☐1.5 ☐2

.....

.....

.....

.....

.....

Otázka 10 Uvažujte následující příkaz BASHe:

```
l=`ps`
```

Který z následujících příkazů vytiskne na každou řádku jeden proces?

☐ echo $l ☐ echo '$l' ☐ echo "$l" ☐ echo ${l} ☐ echo ($l)

Otázka 11 Vlákna na jedno-procesorovém počítači (bez hyper-threadingu):

- ☐ mohou běžet paralelně pouze pokud použijí mutex
- ☐ mohou běžet paralelně pokud nikdy nepoužívají sdílené proměnné
- ☐ mohou běžet paralelně
- ☐ nemohou běžet paralelně nikdy

Otázka 12 ♣ Jaký je vztah pojmů „page fault“ (výpadek stránky) a „segmentation fault“

- ☐ Dva různé pojmy pro stejnou věc
- ☐ Page fault běžně nastává během běhu většiny programů
- ☐ Segmentation fault je vždy důsledek page fault
- ☐ Page fault se používá k implementaci copy-on-write
- ☐ Segmentation fault běžně nastává během běhu většiny programů
- ☐ Page fault je vždy důsledek segmentation fault
- ☐ *Nic z výše uvedeného není správně*

Otázka 13 Co provádí následující příkaz? tr -d ’t’

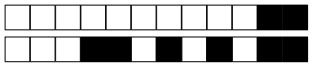
- ☐ Změní všechny znaky d na znak t
- ☐ Zkopíruje standardní vstup beze změny
- ☐ Odstraní tabulátory ze standardního vstupu
- ☐ Odstraní znaky t ze standardního vstupu

Otázka 14 ♣ V roce 2018 zveřejněná zranitelnost CPU zvaná Meltdown:

- ☐ způsobí, že při vykonání škodlivého kódu se procesor přehřeje a dojde k jeho zničení
- ☐ vyskytuje se pouze u procesorů od Intelu
- ☐ je jen další mediální bublinou
- ☐ umožňuje číst data i ze stránek, ke kterým nemá uživatelský proces přístup
- ☐ je to chyba HW a OS s tím nemůže nic dělat
- ☐ *Nic z výše uvedeného není správně*

Otázka 15 ♣ Která následující tvrzení jsou pravdivá?

- ☐ Mutex lze použít pro synchronizaci paralelně běžících vláken
- ☐ Při použití mutexu může vždy nastat deadlock – vzájemné uváznutí
- ☐ Mutex zaručuje, že nenastane deadlock – vzájemné uváznutí
- ☐ Mutex lze použít pro správný přístup ke sdíleným datům
- ☐ *Nic z výše uvedeného není správně*



+4/1/42+

Ukázka zkouškové písemky OSY

Jméno a příjmení:
.....

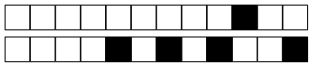
Odpovězte na otázky zaškrtnutím příslušného políčka. Otázky označené znakem ♣ mohou mít více než jednu správnou odpověď. U otázek se slovní odpovědí nezaškrtávejte políčka na šedém pozadí.

Otázka 1 ♣ V tabulce stránek 32bitového systému x86 je pro virtuální adresu 0x12345678 uvedena hodnota 0xCCCCC005.

```
enum {
  PRESENT = 1<<0,
  RW      = 1<<1,
  USER    = 1<<2,
  ACCESS  = 1<<5,
  DIRTY   = 1<<6,
};
```

Které z následujících tvrzení je pravdivé:

- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC003
- ☐ Jádro OS může z této stránky číst
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC678
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC123
- ☐ Uživatelský proces může z této stránky číst
- ☐ Uživatelský proces může do této stránky zapisovat
- ☐ *Nic z výše uvedeného není správně*



Otázka 2 Mějme následující program:

```
int a[2];
pthread_mutex_t mutex[2];

void *fce1(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[0]);
        pthread_mutex_lock(&mutex[1]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[1]);
        pthread_mutex_unlock(&mutex[0]);
    }
    pthread_exit(NULL);
}

void *fce2(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[1]);
        pthread_mutex_lock(&mutex[0]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[0]);
        pthread_mutex_unlock(&mutex[1]);
    }
    pthread_exit(NULL);
}

int main()
{
    pthread_t tid[2];
    a[0]=0; a[1]=1;
    pthread_mutex_init(&mutex[0], NULL);
    pthread_mutex_init(&mutex[1], NULL);
    pthread_create(&tid[0], NULL, fce1, NULL);
    pthread_create(&tid[1], NULL, fce2, NULL);
    pthread_join(tid[0], NULL);
    pthread_join(tid[1], NULL);
    return 0;
}
```

- ☐ program vždy skončí chybou
- ☐ program někdy skončí, někdy uvázne v deadlocku
- ☐ program neskončí, obsahuje nekonečnou smyčku
- ☐ program vždy bez problémů skončí
- ☐ program vždy uvázne v deadlocku

Otázka 3 ♣ Software šířený pod licencí GPLv2:

- ☐ musí být šířen společně se zdrojovými kódy
- ☐ není možné prodávat za peníze
- ☐ je možné použít v proprietárním SW, ale pouze po odstranění všech hlášek ”copyright”
- ☐ nesmí se používat s komerčními OS (např. MS Windows)
- ☐ *Nic z výše uvedeného není správně*

Otázka 4 ♣ Stárnutí (starvation) je problémem, který hrozí plánovacím algoritmům. Které algoritmy ohrožuje stárnutí:

- ☐ nejkratší proces první – shortes process next
- ☐ cyklické plánování – round robin
- ☐ prioritní plánování
- ☐ First Come First Served – podle pořadí spuštění
- ☐ *Nic z výše uvedeného není správně*

Otázka 5 Uvažujte následující příkaz BASHe:

```
l=`ps`
```

Který z následujících příkazů vytiskne na každou řádku jeden proces?

☐ echo ($l) ☐ echo ${l} ☐ echo ’$l’ ☐ echo ”$l” ☐ echo $l

Otázka 6 ♣ Hardwarově asistovaná virtualizace (např. VT-x)

- ☐ funguje jedině na 32bitovém systému.
- ☐ zrychluje vykonávání některých privilegovaných instrukcí.
- ☐ odstraňuje nutnost emulovat hardware (disk, síťová karta, …) mechanismem trap-and-emulate.
- ☐ přidává další vrstvu stránkovacích tabulek.
- ☐ zrychluje běh virtuálního uživatelského režimu.
- ☐ *Nic z výše uvedeného není správně*

Otázka 7 Provedení funkce z jádra operačního systému na architektuře x86 lze z uživatelského programu vyvolat:

- ☐ instrukcí int s registrem obsahujícím číslo služby jádra
- ☐ instrukcí jmp na adresu služby jádra
- ☐ instrukcí call na adresu služby jádra
- ☐ instrukcí nop s registrem obsahujícím číslo služby jádra

Otázka 8 ♣ K zajištění konzistence souborového systému po náhlém vypnutí či pádu systému

- ☐ je vždy potřeba kompletní kontrola souborového systému po následném zapnutí počítače.
- ☐ je potřeba provádět všechny související modifikace souborového systému atomicky.
- ☐ je potřeba používat RAID1 a vyšší.
- ☐ je potřeba vybavit počítač záložním zdrojem (tzv. UPS).
- ☐ lze použít metodu, kdy se popis potřebných modifikací nejprve uloží do speciální oblasti na disku.
- ☐ *Nic z výše uvedeného není správně*

Otázka 9 Vlákna na jedno-procesorovém počítači (bez hyper-threadingu):

- ☐ mohou běžet paralelně pokud nikdy nepoužívají sdílené proměnné
- ☐ nemohou běžet paralelně nikdy
- ☐ mohou běžet paralelně
- ☐ mohou běžet paralelně pouze pokud použijí mutex

Otázka 10 ♣ Která následující tvrzení jsou pravdivá?

- ☐ Mutex lze použít pro správný přístup ke sdíleným datům
- ☐ Mutex zaručuje, že nenastane deadlock – vzájemné uváznutí
- ☐ Mutex lze použít pro synchronizaci paralelně běžících vláken
- ☐ Při použití mutexu může vždy nastat deadlock – vzájemné uváznutí
- ☐ *Nic z výše uvedeného není správně*

Otázka 11 Co provádí následující příkaz? tr -d 't'

- ☐ Odstraní znaky t ze standardního vstupu
- ☐ Odstraní tabulátory ze standardního vstupu
- ☐ Změní všechny znaky d na znak t
- ☐ Zkopíruje standardní vstup beze změny

Otázka 12 ♣ Jaký je vztah pojmů „page fault" (výpadek stránky) a „segmentation fault"

- ☐ Page fault běžně nastává během běhu většiny programů
- ☐ Page fault se používá k implementaci copy-on-write
- ☐ Page fault je vždy důsledek segmentation fault
- ☐ Segmentation fault běžně nastává během běhu většiny programů
- ☐ Segmentation fault je vždy důsledek page fault
- ☐ Dva různé pojmy pro stejnou věc
- ☐ *Nic z výše uvedeného není správně*

Otázka 13 Předpokládejte, že nedojde k chybě při spuštění následujícího programu:

```
int main() {
  int f=fork();
  f=fork();
  f=fork();
  printf("%d\n", f);
  return 0;
}
```

Jaký bude výstup tohoto programu?

- ☐ osm 0
- ☐ šest nenulových čísel a dvě 0
- ☐ tři nenulová čísla a jedna 0
- ☐ osm nenulových čísel
- ☐ dvě nenulová čísla a dvě 0
- ☐ čtyři nenulová čísla a čtyři 0
- ☐ čtyři nenulová čísla a dvě 0

Otázka 14 ♣ V roce 2018 zveřejněná zranitelnost CPU zvaná Meltdown:

- ☐ je jen další mediální bublinou
- ☐ je to chyba HW a OS s tím nemůže nic dělat
- ☐ způsobí, že při vykonání škodlivého kódu se procesor přehřeje a dojde k jeho zničení
- ☐ umožňuje číst data i ze stránek, ke kterým nemá uživatelský proces přístup
- ☐ vyskytuje se pouze u procesorů od Intelu
- ☐ *Nic z výše uvedeného není správně*

Otázka 15 Popište, jak probíhá odesílání dat z aplikace na síť. Co se s daty děje a jakou roli v tom hraje jádro OS? ☐0 ☐0.5 ☐1 ☐1.5 ☐2

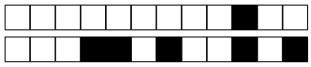
.....

.....

.....

.....

.....



+5/1/36+

Ukázka zkouškové písemky OSY

Jméno a příjmení:
.....

Odpovězte na otázky zaškrtnutím příslušného políčka. Otázky označené znakem ♣ mohou mít více než jednu správnou odpověď. U otázek se slovní odpovědí nezaškrtávejte políčka na šedém pozadí.

Otázka 1 Vlákna na jedno-procesorovém počítači (bez hyper-threadingu):

- ☐ mohou běžet paralelně pokud nikdy nepoužívají sdílené proměnné
- ☐ mohou běžet paralelně
- ☐ mohou běžet paralelně pouze pokud použijí mutex
- ☐ nemohou běžet paralelně nikdy

Otázka 2 ♣ V roce 2018 zveřejněná zranitelnost CPU zvaná Meltdown:

- ☐ způsobí, že při vykonání škodlivého kódu se procesor přehřeje a dojde k jeho zničení
- ☐ umožňuje číst data i ze stránek, ke kterým nemá uživatelský proces přístup
- ☐ vyskytuje se pouze u procesorů od Intelu
- ☐ je jen další mediální bublinou
- ☐ je to chyba HW a OS s tím nemůže nic dělat
- ☐ *Nic z výše uvedeného není správně*

Otázka 3 ♣ Která následující tvrzení jsou pravdivá?

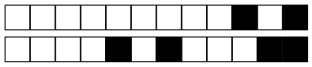
- ☐ Mutex zaručuje, že nenastane deadlock – vzájemné uváznutí
- ☐ Mutex lze použít pro správný přístup ke sdíleným datům
- ☐ Při použití mutexu může vždy nastat deadlock – vzájemné uváznutí
- ☐ Mutex lze použít pro synchronizaci paralelně běžících vláken
- ☐ *Nic z výše uvedeného není správně*

Otázka 4 ♣ Stárnutí (starvation) je problémem, který hrozí plánovacím algoritmům. Které algoritmy ohrožuje stárnutí:

- ☐ cyklické plánování – round robin
- ☐ prioritní plánování
- ☐ First Come First Served – podle pořadí spuštění
- ☐ nejkratší proces první – shortes process next
- ☐ *Nic z výše uvedeného není správně*

Otázka 5 ♣ Jaký je vztah pojmů „page fault“ (výpadek stránky) a „segmentation fault“

- ☐ Segmentation fault je vždy důsledek page fault
- ☐ Page fault se používá k implementaci copy-on-write
- ☐ Page fault běžně nastává během běhu většiny programů
- ☐ Dva různé pojmy pro stejnou věc
- ☐ Page fault je vždy důsledek segmentation fault
- ☐ Segmentation fault běžně nastává během běhu většiny programů
- ☐ *Nic z výše uvedeného není správně*



Otázka 6 ♣ K zajištění konzistence souborového systému po náhlém vypnutí či pádu systému

- ☐ je potřeba vybavit počítač záložním zdrojem (tzv. UPS).
- ☐ je potřeba provádět všechny související modifikace souborového systému atomicky.
- ☐ je potřeba používat RAID1 a vyšší.
- ☐ je vždy potřeba kompletní kontrola souborového systému po následném zapnutí počítače.
- ☐ lze použít metodu, kdy se popis potřebných modifikací nejprve uloží do speciální oblasti na disku.
- ☐ *Nic z výše uvedeného není správně*

Otázka 7 Mějme následující program:

```
int a[2];
pthread_mutex_t mutex[2];

void *fce1(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[0]);
        pthread_mutex_lock(&mutex[1]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[1]);
        pthread_mutex_unlock(&mutex[0]);
    }
    pthread_exit(NULL);
}

void *fce2(void *n) {
    int num=*(int*)n;
    for (int i = 0; i < 150; i++) {
        pthread_mutex_lock(&mutex[1]);
        pthread_mutex_lock(&mutex[0]);
        a[num] += a[1-num];
        pthread_mutex_unlock(&mutex[0]);
        pthread_mutex_unlock(&mutex[1]);
    }
    pthread_exit(NULL);
}

int main()
{
    pthread_t tid[2];
    a[0]=0; a[1]=1;
    pthread_mutex_init(&mutex[0], NULL);
    pthread_mutex_init(&mutex[1], NULL);
    pthread_create(&tid[0], NULL, fce1, NULL);
    pthread_create(&tid[1], NULL, fce2, NULL);
    pthread_join(tid[0], NULL);
    pthread_join(tid[1], NULL);
    return 0;
}
```

- ☐ program neskončí, obsahuje nekonečnou smyčku
- ☐ program vždy bez problémů skončí
- ☐ program někdy skončí, někdy uvázne v deadlocku
- ☐ program vždy uvázne v deadlocku
- ☐ program vždy skončí chybou

Otázka 8 Popište, jak probíhá odesílání dat z aplikace na síť. Co se s daty děje a jakou roli v tom hraje jádro OS? ☐0 ☐0.5 ☐1 ☐1.5 ☐2

.....

.....

.....

.....

.....

Otázka 9 Uvažujte následující příkaz BASHe:

```
l=`ps`
```

Který z následujících příkazů vytiskne na každou řádku jeden proces?

☐ echo ${l} ☐ echo "$l" ☐ echo '$l' ☐ echo ($l) ☐ echo $l

Otázka 10 ♣ Software šířený pod licencí GPLv2:

- ☐ není možné prodávat za peníze
- ☐ nesmí se používat s komerčními OS (např. MS Windows)
- ☐ musí být šířen společně se zdrojovými kódy
- ☐ je možné použít v proprietárním SW, ale pouze po odstranění všech hlášek "copyright"
- ☐ *Nic z výše uvedeného není správně*

Otázka 11 Předpokládejte, že nedojde k chybě při spuštění následujícího programu:

```
int main() {
  int f=fork();
  f=fork();
  f=fork();
  printf("%d\n", f);
  return 0;
}
```

Jaký bude výstup tohoto programu?

- ☐ čtyři nenulová čísla a čtyři 0
- ☐ čtyři nenulová čísla a dvě 0
- ☐ dvě nenulová čísla a dvě 0
- ☐ tři nenulová čísla a jedna 0
- ☐ osm 0
- ☐ osm nenulových čísel
- ☐ šest nenulových čísel a dvě 0

Otázka 12 ♣ Hardwarově asistovaná virtualizace (např. VT-x)

- ☐ funguje jedině na 32bitovém systému.
- ☐ přidává další vrstvu stránkovacích tabulek.
- ☐ zrychluje vykonávání některých privilegovaných instrukcí.
- ☐ odstraňuje nutnost emulovat hardware (disk, síťová karta, …) mechanismem trap-and-emulate.
- ☐ zrychluje běh virtuálního uživatelského režimu.
- ☐ *Nic z výše uvedeného není správně*

Otázka 13 Provedení funkce z jádra operačního systému na architektuře x86 lze z uživatelského programu vyvolat:

- ☐ instrukcí call na adresu služby jádra
- ☐ instrukcí jmp na adresu služby jádra
- ☐ instrukcí int s registrem obsahujícím číslo služby jádra
- ☐ instrukcí nop s registrem obsahujícím číslo služby jádra

Otázka 14 ♣ V tabulce stránek 32bitového systému x86 je pro virtuální adresu 0x12345678 uvedena hodnota 0xCCCCC005.

```
enum {
  PRESENT = 1<<0,
  RW      = 1<<1,
  USER    = 1<<2,
  ACCESS  = 1<<5,
  DIRTY   = 1<<6,
};
```

Které z následujících tvrzení je pravdivé:

- ☐ Uživatelský proces může z této stránky číst
- ☐ Uživatelský proces může do této stránky zapisovat
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC003
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC678
- ☐ Data z této virtuální adresy jsou v RAM na adrese 0xCCCCC123
- ☐ Jádro OS může z této stránky číst
- ☐ *Nic z výše uvedeného není správně*

Otázka 15 Co provádí následující příkaz? tr -d 't'

- ☐ Změní všechny znaky d na znak t
- ☐ Odstraní znaky t ze standardního vstupu
- ☐ Zkopíruje standardní vstup beze změny
- ☐ Odstraní tabulátory ze standardního vstupu