Operativsystem ID1200/06

(ID2200/06 6hp)

Tentamen: exempel på ny struktur

HT19

Instruktioner

- Du får, förutom skrivmateriel, endast ha med dig en egenhändigt handskriven A4 med anteckningar. Anteckningarna lämnas in och kan inte återanvändas.
- Svaren skall lämnas på dessa sidor, använd det utrymme som finns under varje uppgift för att skriva ner ditt svar.
- Svar skall skrivas på svenska eller engelska.
- Du skall lämna in hela denna tentamen och den handskrivna sidan med anteckningar. Inga ytterligare sidor skall lämnas in.

Betyg

Tentamen delas in i olika delar, en grundläggande del bestående av fem frågor och en del för högre betyg också den bestående av fem frågor.

För godkänt, betyg E, skall den grundläggand delen klaras enligt nedan. För högre betyg krävs desutom att två eller flera av de fem övriga frågorna klaras med godkänt resultat.

- Fx: 7 poäng av 10 i den grundläggande delen
- E: 8 poäng av 10 i den grundläggande delen

Högre betyg ges basert på de övriga delarna.

- D: två godkänd
- C: tre godkända
- B: fyra godkända
- A: all godkända

Namn:_____ Persnr:_____

1 stack eller heap [2 poäng]

Vad gör proceduren nedan och var skall `gurka` allokeras, på stacken eller *heapen*? Varför? Skriv färdigt koden så att gurka blir allokerat utrymme.

```
int *tomat(int *a, int *b) {
  // allocate room for gurka


  *gurka = *a + *b;
  return gurka;
}
```

Namn:_____ Persnr:_____

2 fork() [2 poäng]

Vad skrivs ut när vi kör programmet nedan, vilka alternativ finns och varför får vi detta resultat?

```
int global = 17;

int main() {
  int pid = fork();
  if(pid == 0) {
    global++;
  } else {
    global++
    wait(NULL);
    printf("global = %d \n", global);
  }
  return 0;
}
```

Namn:_____ Persnr:_____

3 `__sync_val_compare_and_swap()` [2 poäng]

Vi kan implementera ett spinn-lås i GCC enligt nedan. Låset implementeras med en maskininstruktion som atomärt jämför värdet i en minnesposition och om den uppfyller det krav vi har, ersätter den med ett nytt värde. I implementationen nedan så representerar vi ett öppet lås med värdet 0; om låset är öppet så skriver vi en 1 i den angivna positionen och returnerar 0, i annat fall returnerar vi det funna värdet (som då torde vara 1).

Antag att vi använder låset för att synkronisera två trådar på en maskin med en kärna; vad är då nackdelen som vi kommer ha? Hur skulle vi kunna åtgärda den nackdelen?

```
int try(volatile int *mutex) {
  return __sync_val_compare_and_swap(mutex, 0, 1);
}

void lock(volatile int *mutex) {
  while(try(mutex) != 0) { }
}

void release(volatile int *mutex) {
  *mutex = 0;
}
```

Namn:_____ Persnr:_____

4 a stack, a bottle and ... [2 poäng]

Du har skrivit programmet nedan för att undersöka vad som ligger på stacken.

```
void zot(unsigned long *stop ) {
  unsigned long r = 0x3;
  unsigned long *i;
  for(i = &r;  i <= stop; i++){ printf("%p          0x%lx\n", i, *i);  }
}

void foo(unsigned long *stop ) {
  unsigned long q = 0x2;
  zot(stop);
}

int main() {
  unsigned long p = 0x1;
  foo(&p);
 back:
  printf("  p: %p \n", &p);
  printf("  back: %p \n", &&back);
  return 0;
}
```

Detta är utskriften. Förklara vad som finns på de positioner som pekas ut.

```
0x7ffca03d1748      0x3
0x7ffca03d1750      0x7ffca03d1750
0x7ffca03d1758      0xb93d7906926a7d00
0x7ffca03d1760      0x7ffca03d1790        <-------
0x7ffca03d1768      0x55cdac31d78c        <-------
0x7ffca03d1770      0x7ffca03d17d8
0x7ffca03d1778      0x7ffca03d17b0
0x7ffca03d1780      0x1
0x7ffca03d1788      0x2
0x7ffca03d1790      0x7ffca03d17c0
0x7ffca03d1798      0x55cdac31d7c2
0x7ffca03d17a0      0x55cdac31d810
0x7ffca03d17a8      0x12acac31d5f0
0x7ffca03d17b0      0x1
  p: 0x7ffca03d17b0
  back: 0x55cdac31d7c2
```

Namn:_____ Persnr:_____

5 biblioteksanrop vs systemanrop [2 poäng]

Ett operativsystem som implementerar POSIX skall erbjuda viss funktionalitet för en användarprocess. Tillhandahålls detta genom systemanrop, genom bilbliteksrutiner, eller genom en kombination av de båda. Förklara skillnaden mellan systemanrop och biblioteksrutiner och vad som tillhör operativsystemet.

Namn:_____ Persnr:_____

6 från den ene till ... [P/F]

Antag att vi har två program, `ones` och `add2`, implementerade enligt nedan. Anropet till `scanf("%d", &in)` läser från `stdin` och parsar ett tal som sedan skrivs till `&in`. Proceduren returnerar antingen 1, om den kunde läsa ett tal, eller `EOF`. Anropen till `printf()` kommer skriva ut talen på `stdout`.

```
/* ones.c */
#include <stdlib.h>
#include <stdio.h>

int main() {

  for(int n = 5; n > 0; n--) {
    printf("%d\n", n);
  }

  return 0;
}
```

```
/* add2.c */
#include <stdlib.h>
#include <stdio.h>

int main() {

  int in;
  int result = scanf("%d", &in);

  while(result != EOF) {
    printf("%d\n", in+2);
    result = scanf("%d", &in);
  }

  return 0;
}
```

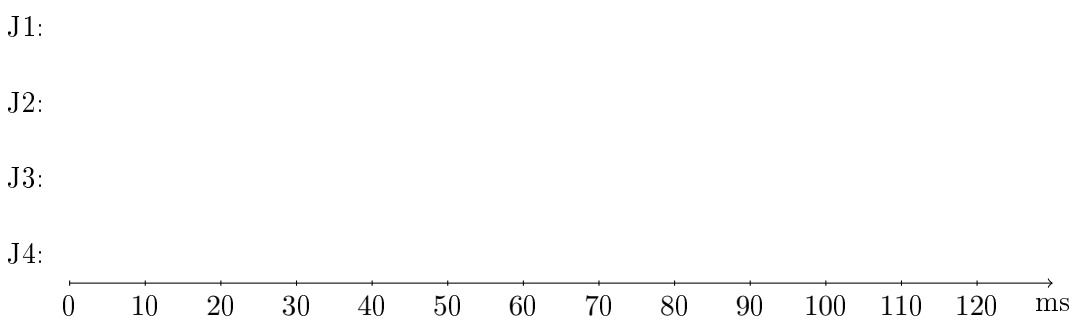
Till ditt förfogande har du en Linux-dator med alla tänkbara program. Hur skulle du på enklast möjliga sätt få utskriften från det ena programmet, `ones`, att läsas av det andra, `add2`.

Namn:_____ Persnr:_____

7 Schemaläggning [P/F]

Antag att vi har en schemaläggare som använder *shortest job first*. Vi har fyra jobb som nedan anges med ⟨*anländer vid*, *exekveringtid*⟩ i ms. Rita upp ett tidsdiagram över exekveringen och ange omloppstiden för vart och ett av jobben.

- J1 : ⟨0,40⟩
- J2 : ⟨0,30⟩
- J3 : ⟨10,10⟩
- J4 : ⟨20,30⟩



Namn:_____ Persnr:_____

8 gungor och karuseller [P/F]

Antag att vi har ett sidindelat virtuellt minne med sidstorlek 4Ki byte. Antag att varje process har fyra segment (t.ex kod, data, stack, extra) och att dessa kan vara av godtycklig men given storlekar. Hur mycket kommer operativsystemet att förlora i intern fragmentering?

Namn:_____ Persnr:_____

9 sidindelat minne med sidor på 64 byte [P/F]

Du har som uppgift att föreslå en arkitektur för en processor som skall arbeta med att sidindelat minne där sidorna är så små som 64 byte. Processorn är en 16-bitars processor och den virtuella adressrymden skall vara 2^{16} byte.

Föreslå ett schema som använder sig av en hierarkisk sidtabell baserad på sidor om 64 byte och förklara hur adressöversättning går till.

Namn:_____ Persnr:_____

10 loggbaserade fs [P/F]

I ett loggbaserar filsystem skriver vi alla förändringar i en kontinuerlig logg utan att göra förändringar i de redan existerande block som en fil har. Vi kommer förr eller senare att få ont om nya block och måste på något sätt återanvända block som inte längre används.

Hur håller vi reda på vilka block som kan återanvändas och hur går vi till väga?