Popis:
Vytvořte jednoduchý shell, který bude implementovat následující funkce:

- spuštění libovolného programu s libovolnými parametry
- při spuštění se bere v úvahu proměnná `PATH` (nastavená před spuštěním shellu)
- spuštění na pozadí pomocí &
- přesměrování vstupu a výstupu pomocí < a >

Po spuštění shell vypíše prompt a bude čekat na vstup pomocí read().

Použijte funkce/volání `fork(), wait(), exec??(), read(), write(), open(), close(), dup()` případně další potřebná. Program bude POSIXový a funkční na všech požadovaných systémech. Není třeba dělat složitější parsing příkazové řádky, stačí reagovat na symboly &, < a >. Vše ostatní je možné (vhodně!) předávat volanému programu. Ukončení se provede na příkaz exit jako u běžného shellu. Pomocí Ctrl-C se ukončí aktuálně běžící proces na popředí (pokud nějaký je). Max. délka vstupu bude 512 znaků. Čtěte 513 znaků - read vám vrátí kolik jich bylo načteno. Pokud bude míň jak 513 vstup zpracujete, jinak vypíšete chybu (příliš dlouhý vstup) a vstup ignorujete (ale je potřeba ho načíst až do konce řádku, jinak se vám zbytek přes 512 znaků objeví jakoby v dalším řádku!). Kontrolu na ukončení synů (běžících na pozadí) provádějte pomocí signálu SIGCHLD.

Shell bude používat dvě vlákna. Jedno pro získání vstupu a druhé pro spouštění příkazů. Synchronizace mezi nimi (bufferu pro uložení příkazu) bude provedena pomocí monitoru implementovaného pomocí posixových mutexů a podmíněných proměnných.

1. vlákno:

```
while (not end) {
   read(stdin) -> buffer
   signal(cond)
}
```

2. vlákno:

```
while (not end) {
   cond_wait(cond)
   zpracuj buffer a proved prikaz
}
```

Pokud vám něco nebude jasné, ptejte se mailem.

Použití může vypadat následovně:

```
$ ls -l -a
$ sleep 10m &
$ echo karel >/tmp/test
```