# Požiadavky na semestrálne práce z predmetu princípy operačných systémov

## Všeobecné pravidlá

- Každú semestrálnu prácu vypracováva skupina 2 (vo výnimočných prípadoch 3) študentov. **Skupina pracuje na semestrálnej práci samostatne.**
- **Originalita každej semestrálnej práce bude kontrolovaná (aj s minuloročnými semestrálnymi prácami).**
- Každá skupina musí odovzdať semestrálnu prácu prostredníctvom aktivity na moodli.
- Semestrálnu prácu musí obhajovať celá skupina.

## Všeobecné požiadavky

Cieľom semestrálnej práce je vytvoriť sieťovú aplikáciu s využitím socketov. V semestrálnej práci:

- musia byť použité **sockety** na medziprocesnú komunikáciu,
- musia byť použité **vlákna**,
- musí byť riešený **synchronizačný problém**,

Ďalej musí semestrálna práca spĺňať nasledujúce požiadavky:

- implementácia v jazyku C/C++,
- návrh v súlade s **princípmi objektového programovanie** (hlavne zapuzdrenie a modularita),
- žiadne úniky pamäte (toto je nutné explicitne ukázať s využitím nástrojov, ktoré sú na to určené),
- kód musí byť logicky rozčlenený do .h a .c/.cpp súborov,
- semestrálna práca musí obsahovať aj **vlastný jednoduchý Makefile**, pomocou ktorého bude možné zostaviť jednotlivé časti semestrálnej práce (napr. klient a server). Makefile musí obsahovať **minimálne nasledujúce ciele**:
  - server,
  - client,
  - all,
  - clean,
- **pri vývoji musí byť využitý verzionovací systém** (preferovane **GitLab**):
  - v rámci verzionovacieho systému musia byť dokladované aspoň **2 príspevky od každého študenta** a musí byť vykonaný aspoň **1 merge vetví projektu**,
  - pred obhajobou semestrálnej práce je nutné vyučujúcemu, u ktorého budete prácu obhajovať, udeliť prístupové právo **Reporter**,
- so semestrálnou prácou odovzdávate:
  - programátorskú dokumentáciu (podľa zverejnenej šablóny), v ktorej popíšete:
    - štruktúru projektu (s využitím UML),
    - kde a ako ste použili sockety,
    - kde a ako ste použili vlákna,
    - aký synchronizačný problém ste riešili a aké prostriedky ste využili pri jeho riešení,
    - ďalšie kľúčové problémy, ktoré ste v rámci semestrálnej práce riešili,
  - používateľskú dokumentáciu, v ktorej popíšete spustenie a ovládanie jednotlivých častí aplikácie,
  - celá semestrálna práca bude zbalená do jedného .zip súboru, ktorý odovzdáte prostredníctvom aktivity na moodli; v .zip súbore budú **iba**:
    - .h súbory,
    - .c/.cpp súbory,
    - Makefile,
    - **ostatné súbory nevyhnutné pre kompiláciu a spustenie aplikácie (textové, grafické a pod.),**
    - programátorská a používateľská dokumentácia.

# Námety na semestrálne práce

**Hra život s ukladaním vzorov na serveri (https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life) (2 študenti)**. K serveru sa môže pripojiť v ľubovoľnom okamžiku ľubovoľný počet klientov. Samotná hra (simulácia) bude bežať na lokálnom počítači, pričom bude umožňovať:

- vytvoriť svet so zadanými rozmermi,
- vygenerovať náhodné rozloženie živých buniek,
- ručne definovať, ktoré bunky sú živé,
- načítať / uložiť svet z / do lokálneho súboru,
- doprednú aj spätnú simuláciu,
- **zapnúť a vypnúť simuláciu v ľubovoľnom okamžiku**,
- pripojiť sa na server a stiahnuť si z neho vzor,
- uložiť vzor na server.

*Simulácia môže byť implementovaná v grafickom prostredí, umožňovať definovanie typu okolia bunky (von Neumannovo okolie a Mooreovo okolie) a typ bunky, z ktorej bude pozostávať hracia plocha (štvorcová, trojuholníková a šesťuholníková) (+ 1 študent).*

**Langtonov mravec s ukladaním vzorov na serveri (https://en.wikipedia.org/wiki/Langton%27s_ant) (2 študenti)**. K serveru sa môže pripojiť v ľubovoľnom okamžiku ľubovoľný počet klientov. Samotná hra (simulácia) bude bežať na lokálnom počítači, pričom bude umožňovať:

- vytvoriť svet so zadanými rozmermi,
- vygenerovať náhodné rozloženie čiernych polí,
- ručne definovať, ktoré polia sú čierne,
- načítať / uložiť svet z / do lokálneho súboru,
- definovať počet mravcov vo svete a ich logiku:
  - priama:
    - mravec sa na bielom poli otočí o 90° vpravo, pole sa zmení na čierne a mravec sa posunie o jedno pole vpred,
    - mravec sa na čiernom poli otočí o 90° vľavo, pole sa zmení na biele a mravec sa posunie o jedno pole vpred,
  - inverzná:
    - mravec sa na bielom poli otočí o 90° vľavo, pole sa zmení na čierne a mravec sa posunie o jedno pole vpred,
    - mravec sa na čiernom poli otočí o 90° vpravo, pole sa zmení na biele a mravec sa posunie o jedno pole vpred,
- definovať, čo sa stane, ak sa na jednom poli stretnú dva alebo viaceré mravce (zánik všetkých mravcov, ktoré sa stretli; prežije iba jeden mravec; polovica mravcov sa začne správať podľa doplnkovej logiky),
- **zapnúť a vypnúť simuláciu v ľubovoľnom okamžiku**,
- pripojiť sa na server a stiahnuť si z neho vzor,
- uložiť vzor na server.

*Simulácia môže byť implementovaná v grafickom prostredí, umožňovať definovanie farieb jednotlivých mravcov a pravidiel popisujúcich, čo sa stane, ak sa na jednom poli stretnú mravce rôznych farieb (+ 1 študent).*

**Simulácia šírenia požiaru s ukladaním máp na serveri (2 študenti)**. V rámci simulácie sa pracuje s 2D svetom pozostávajúcim z buniek reprezentujúcich 4 rôzne biotopy, ktorými sú les, lúka, skaly a voda. Bunky s biotopmi les a lúka sú horľavé. Ďalej sú definované nasledujúce pravidlá:

- ak na niektorej bunke vypukne požiar, tak na susedných bunkách (uvažujeme *von Neumannovo okolie*) obsahujúcich les alebo lúku vypukne požiar za niektorého z nasledujúcich predpokladov:
  - ak veje vietor od horiacej bunky k tejto bunke, tak požiar vznikne s pravdepodobnosťou 90 %,
  - ak je bezvetrie, tak požiar vznikne s pravdepodobnosťou 20 %,
  - ak veje vietor od tejto bunky k horiacej, tak požiar vznikne s pravdepodobnosťou 2 %,
- ak je bunka zhorená a v jej von Neumannovom okolí sa nachádza bunka predstavujúca vodu, tak s pravdepodobnosťou 10 % sa zmení na lúku,
- ak bunka predstavuje lúku a v jej von Neumannovom okolí sa nachádza bunka predstavujúca les, tak s pravdepodobnosťou 2 % sa zmení na les,
- počas simulácie sa v celom svete môže vyskytovať vietor, ktorého smer musí byť zobrazený v simulácii. Vietor má na všetkých bunkách rovnaký smer, pričom platí:
  - bezvetrie sa vyskytuje s pravdepodobnosťou 90 %,
  - vietor sa vyskytuje s pravdepodobnosťou 10 %, pričom jeho smer je rovnaký v celom svete. Ak sa vietor vyskytne, tak bude viať aspoň počas 3 kôl simulácie rovnakým smerom,

- v ľubovoľnom okamžiku simulácie je možné definovať jednu alebo viacero buniek, na ktorých práve vypukol požiar.

Úlohou je vytvoriť aplikáciu pozostávajúcu z klientskej a serverovej časti. Klientska časť bude zodpovedná za beh a ovládanie simulácie, serverová časť bude slúžiť na uchovávanie máp 2D svetov. V rámci klientskej časti, v ktorej bude bežať simulácia, bude možné:

- vytvoriť svet so zadanými rozmermi – biotop jednotlivých buniek bude možné vygenerovať náhodne (podľa vopred zadaného rozdelenia pravdepodobnosti) alebo ručne,
- načítať / uložiť svet z / do lokálneho súboru,
- **zapnúť a vypnúť simuláciu v ľubovoľnom okamžiku,**
- v ľubovoľnom okamžiku definovať, ktoré bunky budú horieť,
- pripojiť sa na server a stiahnuť si z neho príslušný svet,
- uložiť aktuálny svet na server.

*Simulácia môže byť implementovaná v grafickom prostredí a umožňovať definovanie farieb jednotlivých biotopov (+ 1 študent).*

**Sieťová hra pre dvoch alebo viacerých hráčov (2 študenti)**. Všeobecné požiadavky:

- jeden hráč vystupuje ako server, ostatní hráči sa k nemu pripoja,
- hra sa spustí, keď sa pripoja všetci hráči,
- hra môže skončiť podľa definovaných pravidiel (závisia od konkrétnej hry), alebo keď sa od nej odpoja všetci hráči,
- po skončení hry sa danému hráčovi zobrazí jeho výsledok.

Námety na hry:

- hadík,
- bulanci,
- človeče nehnevaj sa,
- pong,
- iné.

Hra môže byť implementovaná v textovom režime (s využitím funkcií zo stdio.h, alebo v prípade Linuxu s využitím funkcií z ncurses.h) alebo v grafickom režime (napr. knižnica SDL).

**Synchronizačný nástroj (2 študenti)**. Aplikácia, ktorá umožní používateľovi:

- špecifikovať typy súborov, ktoré sa majú alebo nemajú synchronizovať,
- automaticky alebo na vyžiadanie synchronizovať súbory a adresáre medzi dvomi adresármi, z ktorých:
  - oba sú na lokálnom počítači,
  - jeden je na lokálnom počítači a druhý sa nachádza na vzdialenom stroji (komunikácia so vzdialeným strojom musí byť šifrovaná),
- obnoviť zmazané súbory, ktoré nie sú staršie ako stanovaný počet dní.

**Jednoduchý databázový systém (2 študenti)**. Systém beží na serveri a môže obsahovať ľubovoľný počet tabuliek. K serveru sa môže pripojiť ľubovoľný počet používateľov, ktorí si môžu:

- vytvoriť účet na serveri a prostredníctvom neho sa prihlásiť,
- vytvoriť tabuľku a definovať jej štruktúru:
  - stĺpce môžu byť typu string, int, double, boolean, date,
  - stĺpce môžu byť null alebo not null,
  - každá tabuľka má definovaný primárny kľúč,
- zrušiť tabuľku (tabuľku môže zrušiť len ten používateľ, ktorý ju vytvoril),
- prideľovať práva ostatným používateľom (právo na vkladanie / aktualizovanie / vypisovanie / mazanie záznamov),
- ak má používateľ príslušné práva:
  - pridávať záznamy (podľa primárneho kľúča),
  - aktualizovať záznamy (podľa zadaných podmienok),
  - mazať záznamy (všetky alebo podľa zadaných podmienok),
  - vypísať záznamy v neutriedenom poradí (všetky alebo podľa zadaných podmienok),
  - vypísať záznamy v utriedenom poradí podľa zvoleného stĺpca (všetky alebo podľa zadaných podmienok),
- vypísať zoznam tabuliek, ktoré vytvorili,
- vypísať zoznam tabuliek, ku ktorým majú príslušné práva.

S jednou tabuľkou môže v danom okamžiku pracovať ľubovoľný počet používateľov. Všetky tabuľky musia byť riešené vo forme perzistentných záznamov tak, aby sa k nim dalo dostať aj po ukončení aplikácie, t. j. **databáza nemôže byť implementovaná iba v operačnej pamäti**.

**Anonymizačná sieť (2 študenti)**. Anonymizačná sieť slúži na skrytie identity používateľov, ktorí sa cez ňu pripájajú do internetu. Pozostáva z niekoľkých uzlov, cez ktoré je smerovaná komunikácia medzi používateľom a zvolenou službou internetu. Príkladom je sieť Tor (https://en.wikipedia.org/wiki/Tor_(anonymity_network)). V rámci semestrálnej práce je nutné vytvoriť klient-server aplikáciu, kde server bude:

- uchovávať informácie o uzloch anonymizačnej siete,

a klientska aplikácia umožní používateľovi:

- pripojiť sa do anonymizačnej siete:
  - ako koncový používateľ (používateľ zadá počet uzlov, cez ktoré sa má smerovať komunikácia a tie sa následne náhodne vyberú zo zoznamu uzlov dostupných na serveri),
  - ako uzol siete (musia byť aktualizované informácie na serveri),
- odpojiť sa od anonymizačnej siete (musia byť aktualizované informácie na serveri),
- stiahnuť obsah zvolenej internetovej stránky (protokoly http a https),
- *šifrovať komunikáciu technikou „onion routing" (https://en.wikipedia.org/wiki/Onion_routing) (+1 študent)*.

**Download manažér (2 študenti)**. Klientska aplikácia, ktorá dovolí používateľovi pripojiť sa na vzdialený server a pomocou štandardných protokolov stiahnuť obsah dostupný na serveri. Manažér musí podporovať nasledujúce protokoly:

- http, https, ftp, ftps,
- *bittorent (+ 1 študent)*,

pričom umožní:

- naplánovať čas, kedy sa má sťahovanie začať,
- ukončiť sťahovanie,
- spravovať lokálne adresáre,
- spravovať históriu sťahovania,
- definovať prioritu sťahovania,
- v prípade súčasného sťahovania viacerých súborov ich naozaj sťahovať súčasne a nie postupne.

Pri protokoloch, ktoré to dovoľujú, aplikácia tiež umožní:

- pozastaviť alebo obnoviť sťahovanie súboru.

***Poznámka: aplikácia nesmie byť implementovaná s použitím nástroja Wget a iných podobných nástrojov slúžiacich na sťahovanie internetového obsahu.***