Semestrálna práca 1 – analýza výkonu údajových štruktúr

Úlohou prvej semestrálnej práce je experimentálne porovnať výkon rozdielnych implementácií vybraných ADT. Výstupom semestrálnej práce bude:

1. knižnica Vami testovaných údajových štruktúr (podľa zvolenej úrovne semestrálnej práce);
2. aplikácia schopná otestovať štruktúry a vyprodukovať výsledné CSV súbory;
3. CSV súbory s výstupmi z testov;
4. dokumentácia, ktorá bude obsahovať (prvé 4 body dokumentácie pre každý test sú podrobne charakterizované v rámci kontroly rozpracovania semestrálnej práce, ktorej popis sa nachádza na konci tohto zadania):
   - UML diagram návrhu univerzálnych testov a aplikácie,
   - popis realizácie priebehu jednotlivých scenárov v testoch,
   - popis formátu údajov a CSV súborov v testoch,
   - metodiku spracovania a vyhodnotenia výsledkov testov (CSV súborov),
   - prezentáciu výsledkov a záverov vyplývajúcich z testovania.

Údajové štruktúry musia byť Vami naprogramované, správne objektovo navrhnuté, univerzálne, a **efektívne** implementované z pohľadu výpočtovej a pamäťovej zložitosti. **V kóde používajte namiesto číselných konštánt symbolické.** Údajové štruktúry a testovacia aplikácia musia byť naprogramované v **jazyku s manuálnou správou pamäte (t. j. bez garbage collector-u)**.

Testovacia aplikácia bude slúžiť na testovanie štruktúr a vyprodukovanie (a prípadné spracovanie) výstupu. Musí obsahovať intuitívne používateľské rozhranie (konzola alebo grafické rozhranie). Aplikácia umožní zvoliť testovanú štruktúru a spustiť množinu testov. Testy vyprodukujú CSV súbor s ich výsledkami. Po ukončení behu aplikácie musí byť pamäť **preukázateľne čistá** (nevznikli „memory leak-y“). **Testy musia byť naprogramované univerzálne. Vstup do testov bude ADT štruktúry** (napr. `List<T>`, nie `ArrayList<T>` ani `LinkedList<T>`)!

CSV (comma-separated values) súbor je textový súbor obsahujúci údaje oddelené čiarkami. Takéto súbory je možné ďalej jednoducho spracovávať tabuľkovými editormi (napr. MS Excel). Údaje v CSV súboroch sú jednoducho prenositeľné medzi aplikáciami. Štruktúra súboru nie je nijako pevne daná (teda, koľko musí mať riadkov, koľko údajov je v jednotlivých riadkoch atď.). Význam jednotlivým riadkom/stĺpcom je potrebné určiť a zdokumentovať v dokumentácii semestrálnej práce. Viac o CSV súboroch sa dočítate napr. na https://datahub.io/docs/data-packages/csv.

Vyprodukované CSV súbory je potrebné spracovať (akokoľvek – ručne, pomocou tabuľkového editora, pomocou ďalšej alebo tej istej aplikácie). **Metodické popísanie spôsobu spracovania údajov je súčasťou dokumentácie semestrálnej práce.** Pomocou popísaného postupu musí byť možné nové údaje spracovať rovnako, ako pôvodné. Výsledky spracovania (grafy, priemery, odhady, atď.) je potrebné zdokumentovať a interpretovať (vyvodiť závery). Identifikácia relevantných veličín je súčasťou semestrálnej práce.

(5b) ADT zoznam

Cieľom je otestovať časovú zložitosť jednotlivých operácií ADT zoznam v závislosti od aktuálneho počtu prvkov v danom ADT a od jeho konkrétnej implementácie.

ADT zoznam musí podporovať operácie definované v Tab. 1 (názvy operácií v implementácii semestrálnej práce môžu súhlasiť s názvami operácií implementovanými v aplikácii AUS používanej na cvičeniach).

Tab. 1 Operácie podporované ADT zoznam

Operácia	Parametre	Návratová hodnota
Vytvor		prázdny zoznam
Zruš		
Vlož prvý	prvok	
Vlož posledný	prvok	
Vlož na index	prvok, index	
Zruš prvý		prvok
Zruš posledný		prvok
Zruš na indexe	index	prvok
Sprístupni	index	prvok
Nastav	index, prvok	
Index prvku	prvok	index

V scenároch definovaných v Tab. 2 porovnajte výkon implementácie:

- ***poľom*** (`ArrayList`),
- ***zreťazenou pamäťou*** (`LinkedList`).

Tab. 2 Testovacie scenáre pre ADT zoznam

Scenár	Podiel operácií			
	Vlož prvý *Vlož posledný* *Vlož na index*	*Zruš prvý* *Zruš posledný* *Zruš na indexe*	*Sprístupni* *Nastav*	*Index prvku*
A	20	20	50	10
B	35	35	20	10
C	45	45	5	5

V každom scenári vykonajte spolu **100 000** operácií. Jednotlivé operácie sú v jednotlivých scenároch volané náhodne tak, aby na konci súhlasil podiel jednotlivých operácií (neplatí, že najskôr sa zavolajú operácie `Vlož`, potom operácie `Zruš`, potom operácie `Sprístupni` resp. `Nastav` a nakoniec operácia `Index prvku`). Ak je možné vybrať z viacerých operácií, operácie sa vyberú s rovnakou pravdepodobnosťou. Parametre do operácií sú taktiež náhodné; ako index je možné zvoliť akýkoľvek aktuálne platný index.

(5b) ADT prioritný front

Cieľom je otestovať časovú zložitosť jednotlivých operácií ADT prioritný front v závislosti od aktuálneho počtu prvkov v danom ADT a od jeho konkrétnej implementácie.

ADT prioritný front musí podporovať operácie definované v Tab. 3 (názvy operácií v implementácii semestrálnej práce môžu súhlasiť s názvami operácií implementovanými v aplikácii AUS používanej na cvičeniach).

Tab. 3 Operácie podporované ADT prioritný front

Operácia	Parametre	Návratová hodnota
Vytvor		prázdny prioritný front
Zruš		
Vlož	prvok, priorita	
Vyber		prvok s najvyššou prioritou
Ukáž		prvok s najvyššou prioritou

V scenároch definovaných v Tab. 4 porovnajte výkon implementácie:

- ***zoznamom implementovaným poľom, ktoré je utriedené podľa priorít,***
- ***ľavostrannou haldou.***

Tab. 4 Testovacie scenáre pre ADT prioritný front

Scenár	Podiel operácií		
	Vlož	*Vyber*	*Ukáž*
A	35	35	30
B	50	30	20
C	70	25	5

V každom scenári vykonajte celkovo **100 000** operácií. Jednotlivé operácie sú v jednotlivých scenároch volané náhodne tak, aby na konci súhlasil podiel jednotlivých operácií (neplatí, že najskôr sa zavolajú operácie `Vlož`, potom operácie `Vyber` nakoniec operácia `Ukáž`). Parametre do operácií sú taktiež náhodné; priorita je náhodné číslo z intervalu <0; 10 000>.

(10b) ADT viacrozmerné pole – matica

Cieľom je otestovať časovú zložitosť algoritmov pracujúcich s ADT viacrozmerné pole – matica v závislosti od parametrov algoritmov a použitej implementácie daného ADT.

Preskúmajte výkon rôznych implementácií ADT viacrozmerné pole – matica v prípade jednoduchých maticových algoritmov. Matica musí podporovať operácie definované v Tab. 5 (názvy operácií v implementácii semestrálnej práce môžu súhlasiť s názvami operácií implementovanými v aplikácii AUS používanej na cvičeniach).

Tab. 5 Operácie podporované ADT viacrozmerné pole – matica

Operácia	Parametre	Návratová hodnota
Vytvor	počet riadkov, počet stĺpcov, inicializačná hodnota	matica, ktorej všetky položky sú nastavené na inicializačnú hodnotu
Zruš		
Sprístupni	index riadku, index stĺpca	prvok
Nastav	index riadku, index stĺpca, prvok	

V scenároch definovaných v Tab. 6 porovnajte výkon implementácie:

- ***v súvislej pamäti,***
- ***v nesúvislej pamäti (pole polí).***

Tab. 6 Scenáre pre testovanie ADT viacrozmerné pole – matica

Scenár	Popis scenára
A	Vygenerujte dve obdĺžnikové matice rovnakých rozmerov (m x n) a vypočítajte tretiu maticu (rozmerov m x n), ktorá bude ich súčtom.
B	Vygenerujte dve obdĺžnikové matice rozmerov m x n a n x m a vypočítajte tretiu maticu (rozmerov m x m), ktorá bude ich súčinom.

Cieľom experimentov je odhadnúť vplyv implementácie a rozmerov matíc (parametre m a n) na algoritmus sčitovania matíc (A) a algoritmus násobenia matíc (B). Za týmto účelom je nutné meniť hodnoty parametrov m a n tak, aby ste zistili, ktorý z týchto parametrov má väčší vplyv v jednotlivých algoritmoch. Hodnotu každého z týchto parametrov definujte aspoň z rozsahu **1 .. 2 000**. Na základe vykonaných experimentov odhadnite hornú asymptotickú zložitosť algoritmu sčitovania matíc (scenár A) a algoritmu násobenia matíc (scenár B) a zdôvodnite, ktorá z dvoch testovaných implementácií ADT viacrozmerné pole – matica je rýchlejšia.

(20b) Testovanie ďalších štruktúr

Cieľom je otestovať časovú zložitosť nižšie uvedených implementácií vybraných ADT v závislosti od aktuálneho počtu prvkov v danej implementácii ADT a jej parametrov (ak sú definované v príslušnom popise).

- **(5b) Obojstranne zreťazený cyklický zoznam** – implementácia musí podporovať funkcionalitu ADT zoznam uvedenú v Tab. 1. Na testovanie použite scenáre definované v Tab. 2.
- **(10b) Dvojzoznam ako implementácia prioritného frontu** – otestujte vplyv dĺžky kratšieho zoznamu na výkon štruktúry (implementácia prioritného frontu ako dvojzoznam musí podporovať funkcionalitu ADT prioritný front uvedenú v Tab. 3). Pri testovaní je nutné porovnať nasledujúce stratégie pre definovanie dĺžky kratšieho zoznamu:
  - konštantná dĺžka definovaná ako 1/1000 počtu všetkých vložení do prioritného frontu;
  - variabilná dĺžka definovaná ako $\sqrt{n}$, kde n je počet prvkov v prioritnom fronte;
  - variabilná dĺžka definovaná ako $n/2$, kde n je počet prvkov v prioritnom fronte.

  Uvedené stratégie porovnajte s využitím scenárov definovaných v Tab. 4.
- **(10b) Množina ako bitová mapa** – otestujte vplyv veľkosti bázovej množiny na rýchlosť jednotlivých operácií. Množina musí podporovať funkcionalitu uvedenú v Tab. 7 (názvy operácií v implementácii semestrálnej práce môžu súhlasiť s názvami operácií implementovanými v aplikácii AUS používanej na cvičeniach). Vhodne navrhnite reprezentáciu bázovej množiny prvkov. Množina nemusí byť implementovaná univerzálne. Stačí, ak vytvoríte množinu ordinálneho typu (napr. int), pričom bázová množina bude definovaná minimálnou (*min*) a maximálnou hodnotou (*max*), ktorú príslušný ordinálny typ môže nadobúdať. V takomto prípade môžete samotnú množinu (podmnožinu bázovej množiny) implementovať ako pole obsahujúce aspoň ***max* - *min* + 1 bitov** (**nie bajtov**). Hodnota *i*-tého bitu (pre $i = 0,1,..., max - min$) potom indikuje, či prvok $min + i$ bázovej množiny sa nachádza v množine (hodnota 1) alebo nenachádza (hodnota 0). Napr., ak bude váš ordinálny typ int, pričom *min* bázovej množiny bude -100 a *max* 1 000, tak na reprezentáciu množiny (podmnožiny bázovej množiny) budete potrebovať 1 101 bitov. Hodnota 0-tého bitu bude hovoriť, či sa prvok -100 nachádza v množine, hodnota 1-ého bitu, či sa tam nachádza prvok -99, hodnota 2-ého bitu, či tam je prvok -98, atď. až hodnota 1 100-tého bitu bude korešpondovať s prítomnosťou prvku 1 000 v množine. V tomto prípade pole, ktoré bude obsahovať požadovaný počet bitov, môžete implementovať ako pole bajtov s veľkosťou 1101/8 +1 = 137 +1 = 138 bajtov. Množinové operácie, akými sú prienik a zjednotenie, by ste museli implementovať ako prechod bajtmi tohto poľa a druhého rovnako veľkého poľa reprezentujúceho druhú množinu a aplikovaním príslušnej bitovej operácie (AND alebo OR) na jednotlivé prvky oboch polí. V takomto prípade by ste potrebovali vypočítať príslušnú bitovú operáciu 138-krát. Výrazne lepšie riešenie spočíva vo využití poľa prvkov typu unsigned long long (na 64-bitových architektúrach). V takomto prípade budete potrebovať na reprezentáciu 1 101 bitov pole, ktoré bude mať 1101/64 +1 = 18 (64-bitových) položiek. Vďaka tomu

bude výpočet príslušnej bitovej operácie vyžadovať prechod poľami, ktoré majú iba 18 položiek. Preto tento spôsob realizácie množiny predstavuje výrazne lepšie riešenie ako predchádzajúci prípad.

Vplyv veľkosti bázovej množiny na každú z operácií definovaných v Tab. 7 (s výnimkou operácií *Vytvor* a *Zruš*) preskúmajte prostredníctvom **dvoch vami navrhnutých testovacích scenárov**, pričom v každom scenári sa musí meniť veľkosť bázovej množiny aspoň z rozsahu **10 .. 100 000**. V rámci dokumentácie je potrebné popísať povahu generovaných údajov, ako aj každý testovací scenár (buď uviesť celkový počet operácií vrátane podielov jednotlivých operácií, alebo popísať konkrétnu úlohu, ktorú použijete ako testovací scenár).

Tab. 7 Operácie podporované ADT množina

Operácia	Parametre	Návratová hodnota
Vytvor	bázová množina	prázdna množina
Zruš		
Vlož	prvok	
Odober	prvok	
Patrí	prvok	pravda/nepravda
Je rovná	množina	pravda/nepravda
Je podmnožinou	množina	pravda/nepravda
Zjednotenie	množina	nová množina
Prienik	množina	nová množina
Rozdiel	množina	nová množina

Bonus (5b)

Cieľom je navrhnúť univerzálny spôsob zadávania testov ADT a následnej tvorby scenárov vo vašej testovacej aplikácii.

Univerzálny spôsob zadávania testov spočíva v tom, že nebudú naprogramované vopred, ale používateľ ich bude môcť zostaviť. Aplikácia teda musí používateľovi umožniť vytvárať testy, na ich základe tvoriť testovacie scenáre, tie následne uložiť, opätovne načítať a, samozrejme, spustiť. V rámci bonusu je nutné umožniť tvorbu testov pre ľubovoľný ADT definovaním celkového počtu operácií a podielov jednotlivých typov operácií. Tvorba testov definovaním konkrétneho algoritmu, na ktorom by bola implementácia ADT skúmaná, sa v tejto časti semestrálnej práce nevyžaduje.

Bonusovú úlohu je možné riešiť a odovzdať po splnení aspoň prvej úrovne semestrálnej práce (testovanie ADT zoznam).

Bodovanie semestrálnej práce

Počet bodov za semestrálnu prácu	Úroveň	Požadovaná funkcionalita
5	1	Zoznamy
10	2	Úroveň 1 + prioritné fronty
20	3	Úroveň 2 + matice
45	4	Úroveň 3 + ďalšie štruktúry
+5	Bonus	Aspoň úroveň 1 + používateľom zadávané testy

Kontrola rozpracovania semestrálnej práce

Dva týždne pred termínom odovzdania semestrálnej práce je nutné odovzdať prvotnú verziu návrhu semestrálnej práce vo forme dokumentácie. Kontrola semestrálnej práce odhalí možné slabiny návrhu a bude sa tak možné vyhnúť problémom pri implementácii. V rámci tejto dokumentácie je nutné popísať návrh a priebeh testov, formát údajov a súboru a metodiku vyhodnotenia (prvé 4 body dokumentácie semestrálnej práce pre každú úroveň) aspoň pre ADT zoznam (teda pre 1. úroveň semestrálnej práce).

V rámci odovzdanej dokumentácie:

- vytvorte UML diagram tried univerzálneho návrhu testov pre ADT zoznam a popíšte ho;
- popíšte, ako budete realizovať jednotlivé scenáre pri testovaní ADT zoznam. Teda, ako sa budete v danom scenári rozhodovať, ktorú skupinu operácií zvolíte a ktorú operáciu z danej skupiny následne vykonáte;
- popíšte údaje, zaznamenávané počas testov, a formát CSV súboru (teda, čo sa bude v CSV súbore nachádzať a kde), do ktorého budete tieto údaje ukladať;
- popíšte, ako budete analyzovať výsledné CSV súbory.

Odovzdaná dokumentácia nebude finálna dokumentácia semestrálnej práce (tam bude aj popis návrhu aplikácie a prezentácia výsledkov a záverov testov), ale môže (a v ideálnom prípade by mala) tvoriť jej časť. V rámci kontroly rozpracovania môžete vypracovať aj návrh ďalších úrovní. Vo finálnej verzii dokumentácie bude potrebné vytvoriť takýto popis pre všetky vypracované úrovne semestrálnej práce.

Bodovanie rozpracovania semestrálnej práce

Za prvotnú verziu dokumentácie semestrálnej práce odovzdanú v rámci kontroly rozpracovania semestrálnej práce je možné získať **najviac 5 bodov**, ktoré predstavujú body mimo bodovania semestrálnej práce. Hodnotená bude časť dokumentácie, ktorá sa bude viazať k prvej úrovni (ak vypracujete dokumentáciu pre viac úrovní, získate spätnú väzbu, ale tieto časti neovplyvnia získané body ani pozitívne, ani negatívne, no ušetríte si prácu s jej vypracovaním neskôr).

Študenti, ktorí neodovzdajú prvotnú verziu dokumentácie v rámci kontroly rozpracovania semestrálnej práce, získajú z nej, ako aj z celej semestrálnej práce automaticky 0 bodov.