|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

**Дисциплина “Типы и структуры данных”**

**Лабораторный практикум №6**

**по теме: «обработка деревьев, хеш-таблиц»**

Выполнил студент: \_\_*Клименко Алексей Константинович\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*фамилия, имя, отчество*

Группа: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*ИУ7-35Б*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил, к.п.н.: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*подпись, дата*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2020 г.*

**Цель работы**

Получить навыки применения двоичных деревьев, реализовать основные операции над деревьями: обход деревьев, включение и поиск узлов; построить и обработать хеш-таблицы, сравнить эффективность поиска в сбалансированных деревьях, в двоичных деревьях поиска и в хеш-таблицах.

**описание условия задачи**

Создать программу для работы с деревьями и хеш-таблицами.

**Техническое задание**

**исходные данные**

Исходными данными являются целые числа из файла.

*Примеры настройки параметров моделирования:*

|  |  |
| --- | --- |
| **Ввод пользователя** | **Результат** |
| config | переход в режим настроек |
| set a 1 5 | задание граничных значений для времени прихода заявки в первую очередь |
| set d 2.3 2.4 | задание граничных значений для времени обработки заявки из второй очереди |
| back | выход из режима настроек |

**результат**

Результатом работы программы является моделирование системы из двух очередей с абсолютным приоритетом и повторным обслуживанием, и обслуживающего аппарата.

**способы обращения к программе**

Для запуска программы необходимо запустить файл **app.exe**. Далее запустить моделирование командой **run A A** или перейти в меню настроек для изменения параметров моделирования.

**возможные аварийные ситуации и ошибки пользователя**

При неверном вводе команды программа попросит ввести команду снова, а при невозможности выполнения команды при неверных параметрах моделирования сообщает об этом, и предлагает задать другие параметры моделирования.

**Структуры данных**

Для реализации очереди на массиве была выбрана следующая структура:

**struct queue\_arr**

**{**

**uint32\_t capacity;** // вместимость очереди

**uint32\_t size;** // размер очереди

**int32\_t \*begin;** // начало массива (нелувой элемент)

**int32\_t \*end;** // указатель за последним элементом в массиве

**int32\_t \*first;** // указатель на первый элемент в очереди

**int32\_t \*last;** // указатель за последним элементом в очереди

**};**

Реализация очереди на связном списке:

**struct \_\_qu\_lst\_node**

**{**

**struct \_\_qu\_lst\_node \*next;** // указатель на следующий узел в очереди

**qdata\_t data;** // данные элемента очереди

**};**

**struct queue\_lst**

**{**

**uint32\_t size;**  // размер очереди

**struct \_\_qu\_lst\_node \*first;** // указатель на первый элемент в очереди

**struct \_\_qu\_lst\_node \*\*last;** // указатель на поле <next>

// в последнем элементе очереди

**};**

Сравнительная таблица эффективности по памяти обоих решений:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| макс. размер | реал. размер | заполнен-ность, % | размер очереди на массиве | размер очереди на списке | эффектив-ность очереди на списке, % |
| 10 | 4 | 40 | 80 | 88 | -10 |
| 10 | 8 | 80 | 80 | 152 | -90 |
| 100 | 15 | 15 | 440 | 264 | 40 |
| 100 | 35 | 35 | 440 | 584 | -33 |
| 1000 | 240 | 24 | 4040 | 3864 | 4 |
| 1000 | 260 | 26 | 4040 | 4184 | -4 |

Таким образом, можно сделать вывод о том, что использование списков для реализации очереди является эффективным по памяти в случаях, когда реальный размер очереди зачастую будет менее **25%** от максимально возможного.

**Набор функций**

// Создаёт новую очередь заданной вместимости.

**struct queue qu\_create(queue\_imp\_t type, uint32\_t capacity);**

**Описание алгоритма обработки данных**

Моделирование работы обслуживающего аппарата состоит из итерационной обработки событий среди которых приход заявки в очередь и смена обрабатываемой заявки в обслуживающем аппарате.

**К := требуемое кол-во заявок 1го типа**

**пока K > 0:**

**Набор функциональных тестов**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Описание теста** | **Входные данные** | **Выходные данные** |
| 1 | Некорректная команда | a | Сообщение о неверной команде. Возврат в меню |
| 2 | Запуск моделирования с настройками по-умолчанию | run A L | Вывод информации в процессе моделирования на экран |
| 3 | Изменение параметров по-умолчанию | config  set a 1.0 2.0  set b 3 3  show | Вывод измененных параметров на экран |
| 4 | Сброс настроек | config  set a 1.0 2.0  set b 3 3  reset  show | Вывод настроек по-умолчанию |
| 5 | Установка неверного промежутка времени | config  set a 20 18 | Вывод сообщении о неверном временном промежутке |
| 6 | Установка нулевого промежутка времени | config  set a 0 0 | Вывод сообщении о неверном временном промежутке |
| 7 | Переход в ручной режим | manual  show | Вывод пустых очередей |
| 8 | Удаление элемента из пустых очередей | manual  pop | Вывод сообщении о том, что нечего удалять |
| 9 | Превышение максимального числа элементов в очередях | manual  push 1  ...  push 11 | Вывод сообщении о переполнении очередей |

**Тесты эффективности по времени**

Результаты тестирования эффективности по времени операций добавления и удаления из очереди в виде таблицы:

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | время массив | время список | эффективность массива |
| push\_back | 17 тактов | 178 тактов | 90,4 % |
| pop\_front | 18 тактов | 2076 тактов | 99,1 % |

**Выводы по проделанной работе**

Использование той или иной реализации очереди сильно зависит от выполняемых задач. Для представленной задачи моделирования работы обслуживающего аппарата сложно сразу сказать, какая реализация будет наиболее эффективной: у каждой из них есть как достоинства так и недостатки.

Для реализации очереди на массиве характерна быстрота выполняемых операции вставки и удаления. Это решение эффективно по времени всегда, но зачастую проигрывает по памяти реализации на списке, которая показывает свою эффективность при больших скачках размера очереди, но преобладающем малом размере.

**Контрольные вопросы**

1. *Что такое дерево?*

Деверо - это структура данных, описываемая рекурентно как узел, у которого есть указатели на два других узла (левое и правое поддеревья).

2. *Как выделяется память под представление деревьев?*

П

3. *Какие стандартные операции возможны над деревьями?*

Стандартные операции над деревьями включают в себя вставку узла в дерево, поиск узла и удаление.

4. *Что такое дерево двоичного поиска?*

Дерево двоичного поиска - это дерево, в котором для каждого узла задано отношение порядка таким образом, что этот узел меньше одного своего поддерева, но больше другого поддерева.

5. Чем отличается идеально сбалансированное дерево от АВЛ дерева*?*

Идеально сбалансированное дерево определяется как дерево двоичного поиска, в котором у каждого узла ***количество узлов*** в обоих его поддеревьях отличается не более чем на единицу.

В АВЛ деревьях это требование ослаблено. В них у каждого узла ***высоты*** обоих его поддеревьев отличаются не более чем на единицу.

6. *Чем отличается поиск в АВЛ-дереве от поиска в дереве двоичного поиска?*

Поиск в сбалансированном дереве зачастую происходит быстрее, так как

7. *Что такое хеш-таблица, каков принцип ее построения?*

Хеш-таблица это структура данных

8. *Что такое коллизии? Каковы методы их устранения?*

Коллизии - это ситуации, когда для разных ключей выбранная хеш-функция возвращает одно и то же значение.

9. *В каком случае поиск в хеш-таблицах становится неэффективен?*

При

10. *Эффективность поиска в АВЛ деревьях, в дереве двоичного поиска и в хеш-таблицах*

При