МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

Кафедра теории функций и стохастического анализа

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ (БАЗОВОЙ) ПРАКТИКЕ

студента 4 курса 451 группы
направления 38.03.05 — Бизнес-информатика

механико-математического факультета
Чайковского Петра Ильича

Место прохождения: завод "Тантал"
Сроки прохождения: с 29.06.2019 г. по 26.07.2019 г.
Оценка:

Руководитель практики от СГУ

доцент, к. ф.-м. н. _____________ Н. Ю. Агафонова

Руководитель практики от организации

ведущий программист _____________ Д. Э. Кнутов

Саратов 2019

Тема практики:«Правила оформления курсовых и дипломных работ»

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

При рассмотрении таких структур данных, как последовательности и списки, особое внимание привлекает простота их определения: последовательность (список) с базовым типом T - это либо:

1) пустая последовательность (список); либо

2) конкатенация элемента типа T и некоторой последовательности с базовым типом T. Для определения принципов построения, а именно следования или итерации, здесь используется рекурсия. Следования и итерация встречаются настолько часто, что их считают фундаментальными образами строения данных и поведения программ, однако всегда следует помнить, что их можно определять только с помощью рекурсии (обратное неверно), в то время как рекурсии можно эффективно употреблять для определения более сложных структур. Хорошо известным примером служат деревья, которые могут быть определены по следующему принципу: дерево с базовым типом T - это либо:

1) пустое дерево; либо

2) некоторая вершина типа T с конечным числом связанных с ней отдельных деревьев с базовым типом T, называемых *поддеревьями.*

Из сходства рекурсивных определений последовательностей и деревьев ясно, что последовательность (список) есть дерево, в котором каждая вершина имеет не более одного поддерева.

1 Постановка задачи. Описание метода её решения.

Для более корректной постановки задачи сформулируем ещё несколько обязательных определений:

Упорядоченное дерево - это дерево, у которого рёбра (ветви), исходящие из каждой вершины, упорядочены.

Степень вершины - число непосредственных потомков внутренней вершины.

Заметим, что особо важную роль при рассмотрении деревьев играют *упорядоченные* деревья *второй степени.* Такие деревья называются *бинарными.* Определим упорядоченное двоичное дерево как конечное множество элементов (вершин), которое либо пусто, либо состоит из корня (вершины) с двумя отдельными двоичными деревьями, которые называются *левым* и *правым поддеревом* этого корня.

Обращаясь же к проблеме представления деревьев, будем описывать как переменные с фиксированной структурой сами вершины, таким образом, их тип будет зафиксирован, и степень дерева будет определять число ссылочных компонент, указывающих на вершины поддеревьев. Ссылки на пустые деревья будут обозначаться значением *NIL.* Таким образом, компоненты дерева имеют такой вид:

```
TYPE Ptr = POINTER TO Node;
TYPE Node = RECORD op: CHAR;
                left, right: Ptr
              END
```

Листинг 1: Компоненты дерева.

Будем считать, что надо строить дерево минимальной глубины, состоящее из n вершин. В таком случае минимальная высота при заданном числе вершин достигается, если на всех уровнях, кроме последнего, помещается максимальное число вершин. Такого можно добиться, размещая приходящие вершины поровну слева и справа от каждой вершины.

Правило равномерного распределения для известного числа вершин n можно сформулировать, используя рекурсию:

1. Взять одну вершину в качестве корня.

2. Построить тем же способом левое поддерево с nl = n DIV 2 вершинами.

3. Построить тем же способом правое поддерево с nr = n - nl - 1 вершиной.

Такому правилу соответствует следующая процедура построения *идеально сбалансированного дерева* (причём будем исходить от следующего определения: дерево называется *идеально сбалансированным*, если число вершин в его левых и правых поддеревьях отличается не более, чем на 1):

```
PROCEDURE tree(n: INTEGER): Ptr:
  VAR newnode: Ptr;
    x, nl, nr: INTEGER;
 BEGIN
  IF n = 0 THEN newnode := NIL
  ELSE nl := n DIV 2; NR := n - nl - 1;
    ReadInt(x); ALLOCATE(newnode, SIZE(Node));
    WITH newnode DO
      key := x; left := tree(nl); right := tree(nr)
    END
  END
  RETURN newnode
 END tree;
```

Листинг 2: Процедура построения идеально сбалансированного дерева.

2 Вычисление сложности проблемы.

3 Программа решения на языке C++.