|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 1 | | |
| по дисциплине «Структура данных и алгоритмы» | | |
| **СТРУКТУРА ДАННЫХ ТАБЛИЦА** | | |
|  | | |
|  | Бригада 1 | Быков богдан |
| Группа ПМ-22 | Брыкова Дарья |
| Вариант 1)1в | Чёрный антон |
| Преподаватели | Хиценко Валентина Павловна |
|
|  |
|  |
|  |
| Новосибирск, 2023 | | |

1. **Условие**

1. **Анализ задачи**
   1. **Входные данные задачи:**

A ∈ текст на Си  
<текст на Си> ::= <подпрограмма>  
<подпрограмма> ::= <заголовок> <тело>  
<заголовок>  
<тело> := { <предложения> }  
<предложения> ::= <предложение> <предложения> | пусто  
<предложение> ::= <определение переменной> | <оператор>  
<определение переменной> ::= <тип> <имя> | <тип> <имя> = <значение> | <тип> <имя>[<число компонент>] = <значение> | <тип> <имя> [<число компонент>]

<число компонент> ::= целое число  
<значение> ::= целое число  
<тип> ::= int | float | double | bool

* 1. **Выходные данные задачи:**

B ∈ <таблица>  
<таблица> = {<элемент>}  
<элемент> = <имя> <тип> <занимаемая память> <число компонент> <тип компонент> (<элемент> | пусто)  
<имя> = строка  
<тип> = строка  
<занимаемая память> = целое число  
<число компонент> = целое число

<тип компонент> = строка

* 1. **Математическое решение задачи:**

T =

N = размер таблицы T

При a = первая строка А

Повторять

Если в строке а есть переменная, тогда

При k = f(имя переменной) mod N

Повторять

k = (k+1) mod N

Пока ключ T[k] не пустой и не обошли всю таблицу

Если в T[k] есть место, тогда записать информацию о переменной в T[k]

Иначе T – полная

Пока существует a и T не полная

* 1. **Подзадачи**

**Подзадача** «вычисление значения после применения функции хеширования на ключ str».

**Назначение**: вычисление хеш-значения соответствующего str.

**Входные данные**: строка str.

**Выходные данные**: ans – целое число.

**Подзадача** «поиск свободного места в таблице после k для элемента str».

**Назначение:** поиск места в таблице после k, куда можно поместить элемент str.

**Входные данные**: текущий вычисленный ключ k, строка str.

**Выходные данные**: k.

1. **Структуры основных входных и выходных данных**

Внешнее представление таблицы: файл, состоящий из текста программы, в котором могут быть инициализированы переменные, «свойства(имя программного объекта, его тип, размер памяти, требуемой объекту, число компонент и тип компонент), оканчивающегося концом файла.

Внутреннее представление таблицы: таблица с вычисляемым входом (таблица с перемешиванием) с используемым методом деления.

1. **Алгоритм**

**Алгоритм поиска доступной ячейки таблицы**

|  |  |
| --- | --- |
|  |  |
|  | **int find** |
|  | counter = 0; |
|  | while (strcmp(t[k % N].name, "")) { |
|  | k += 1; |
|  | } |
|  | return k; |
|  |  |
|  | **Алгоритм функции хеширования** |
|  | **int h** |
|  | len = strlen(str); |
|  | ans = 0; |
|  | for (i = 0; i < len; i++) |
|  | ans += (str[0]); |
|  | return (ans%N); |
|  |  |
|  |  |
|  | **Алгоритм считывания и заполнения таблицы** |
|  | **void input** |
|  | strcpy\_s(data[0].D, "int"); |
|  | data[0].size = sizeof(int); |
|  | strcpy\_s(data[1].D, "float"); |
|  | data[1].size = sizeof(float); |
|  | strcpy\_s(data[2].D, "double"); |
|  | data[2].size = sizeof(double); |
|  | strcpy\_s(data[3].D, "bool"); |
|  | data[3].size = sizeof(bool); |
|  | fscanf\_s(myfile, "%s", a, sizeof(a)); |
|  | while (strcmp(a, "{")) |
|  | fscanf\_s(myfile, "%s", a, sizeof(a)); |
|  | while (fscanf(myfile, "%s", a) != EOF) { |
|  | for (int i = 0; i < 4; i++) { |
|  | if (!strcmp(a, data[i].D)) { |
|  | strcpy\_s(type, a); |
|  | fscanf\_s(myfile, " %[^[;]", a, sizeof(a)); |
|  | strcpy\_s(name, a); |
|  | key = H(name); |
|  | key = FIND(key, t); |
|  | t[key].mem = data[i].size; |
|  | strcpy\_s(t[key].name, name); |
|  | strcpy\_s(t[key].type, type); |
|  | if (fgetc(myfile) == '[') { |
|  | fscanf\_s(myfile, "%d", &k); |
|  | t[key].k = k;  strcpy\_s(t[key].type, "array");  strcpy\_s(t[key].type\_k, type); |
|  |  |
|  |  |
|  | } |
|  | } |
|  | } |
|  | } |
|  |  |
|  |  |

1. **Структура программы**

Подпрограмма **int find**;

int find(int k, elem\* t)

Параметры: k – число, t – таблица;

Возвращает: k - число.

Подпрограмма **int h**;

int h(char\* str);

Параметры: str - строка;

Возвращает: ans - число.

Подпрограмма «Ввод»;

void input(elem\* t);

Параметры: t – пустая таблица;

Возвращает: t – заполненная таблица.

Подпрограмма «Вывод таблицы»;

void vyvod(elem\* t);

Параметры: t – таблица;

Возвращает: t – таблица.

1. **Текст программы**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

const int N = 19;

FILE\* myfile;

struct elem {

char name[256];

char type[15];

int mem;

int k;

char type\_k[15];

};

struct types {

char D[7];

int size;

};

int find(int k, elem\* t) {

int counter = 0;

while (strcmp(t[k % N].name, "")) {

k += 1;

counter += 1;

if (counter == N) {

return -1;

}

}

return k;

}

int h(char\* str) {

int len = strlen(str);

int ans = 0;

for (int i = 0; i < len; i++)

ans += int(str[0]);

return (ans % N);

}

void vyvod(elem\* t) {

myfile = fopen("output.txt", "w");

for (int i = 0; i < N; i++) {

if (strcmp(t[i].name, "")) {

fprintf(myfile, "%s %s %i %i %s\n", t[i].name, t[i].type, t[i].mem, t[i].k, t[i].type\_k);

}

}

}

void input(elem\* t) {

myfile = fopen("input.txt", "r");

char\* str;

char name[256];

char type[7];

int mem;

int k;

int key = 0;

char type\_k[7];

char a[250];

types data[4];

strcpy\_s(data[0].D, "int");

data[0].size = sizeof(int);

strcpy\_s(data[1].D, "float");

data[1].size = sizeof(float);

strcpy\_s(data[2].D, "double");

data[2].size = sizeof(double);

strcpy\_s(data[3].D, "bool");

data[3].size = sizeof(bool);

fscanf\_s(myfile, "%s", a, sizeof(a));

while (strcmp(a, "{"))

fscanf\_s(myfile, "%s", a, sizeof(a));

while (fscanf(myfile, "%s", a) != EOF && key != -1) {

for (int i = 0; i < 4; i++) {

if (!strcmp(a, data[i].D)) {

key = h(name);

key = find(key, t);

if (key != -1) {

strcpy\_s(type, a);

fscanf\_s(myfile, " %[^[;]", a, sizeof(a));

strcpy\_s(name, a);

t[key].mem = data[i].size;

strcpy\_s(t[key].name, name);

strcpy\_s(t[key].type, type);

if (fgetc(myfile) == '[') {

fscanf\_s(myfile, "%d", &k);

t[key].k = k;

strcpy\_s(t[key].type, "array");

strcpy\_s(t[key].type\_k, type);

}

}

}

}

}

};

int main()

{

elem t[N];

for (int i = 0; i < N; i++) {

strcpy\_s(t[i].name, "");

t[i].k = 0;

strcpy\_s(t[i].type\_k, "");

}

input(t);

vyvod(t);

}

1. **Откладка программы**

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер**  **теста** | **Входные данные** | **Выходные данные** | **Назначение теста** |
| 1 | int main()  {  int a1;  a = a + 1;  } | a1 int 4 0 | Простейший тест |
| 2 | int main()  {  int ab1;  float ba2;  double ca3;  ab1 = ab1 + 1;  bool da4;  int a123[12]  } | ab1 int 4 0  ba2 float 4 0  ca3 double 8 0  da4 bool 1 0  a123 array 4 12 int | Тест на все возможные типы переменных и символьно неоднородные названия |
| 3 | int main()  {  int abc1;  double d3;  d3 = d3 + 1;  float bca1;  } | abc1 int 4 0  d3 double 8 0  bca1 float 4 0 | Тест на программу, когда у переменных будут одинаковые хеш-значения |
| 4 | int main()  {  int a1;  double b1;  float c1;  int a2;  double b2;  float c2;  int a3;  double b3[12];  float c3[2];  bool a4[12];  double b4;  float c4;  int a5[21];  double b5;  float c5;  int a6;  double b6;  float c6;  int a7;  } | a1 int 4 0  b1 double 8 0  b2 double 8 0  c1 float 4 0  c2 float 4 0  a2 int 4 0  a3 int 4 0  b3 array 8 12 double  c3 array 4 2 float  a4 array 1 12 bool  b4 double 8 0  c4 float 4 0  a5 array 4 21 int  b5 double 8 0  c5 float 4 0  a6 int 4 0 | Тест на программ, когда таблица будет заполнена полностью |