Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

Выполнили:

студенты группы 23ВВВ2

Пичаев И.А.

Чупраков С.В.

Принял:

Юрова О. В.

Митрохин М.А.

**ОТЧЕТ**

по лабораторной работе №1

по курсу «Логика и основы алгоритмизации в инженерных

задачах»

на тему «Простые структуры данных»

**Цель работы**

Изучение и повторение простых структур данных.

**Лабораторное задание**

**Задание 1**: написать программу, вычисляющую разницу между максимальным и минимальным элементами массива.

**Задание 2**: написать программу, реализующую инициализацию массива случайными числами.

**Задание 3**: написать программу, реализующую создание массива произвольного размера, вводимого с клавиатуры.

**Задание 4**: написать программу, вычисляющую сумму значений в каждом столбце (или строке) двумерного массива.

**Задание 5**: написать программу, осуществляющую поиск среди структур student структуру с заданными параметрами (фамилией, именем и т.д.).

**Описание метода решения задач**

Пенза 2024

**Задание 1**: Использованы INT\_MIN и INT\_MAX для определения максимального и минимально начальных значений

**Задание 2**: Использована функция rand() для генерации случайных чисел.

**Задание 3**: Использованы функции malloc() для динамического выделения памяти, free() для освобождения выделенной памяти и sizeof() для определения размера элемента массива.

**Задание 4**: Использованы функции malloc() для динамического выделения памяти, free() для освобождения выделенной памяти и sizeof() для определения размера элемента массива.

**Задание 5**: Использована команда struct для создания структуры и typedef для создания типа данных, использованы динамические структуры, функция strcmp для сравнения строк, подключена библиотека string.h для работы со строками.

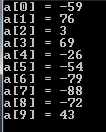
**Результат работы программы**

**Задание 1**:



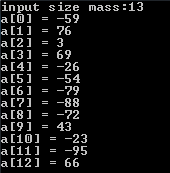
**Рисунок 1 – результат работы программы**

**Задание 2**:

****

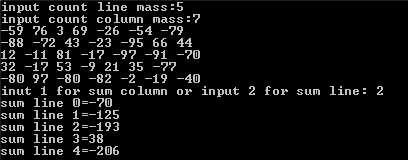
**Рисунок 2 – результат работы программы**

**Задание 3**:



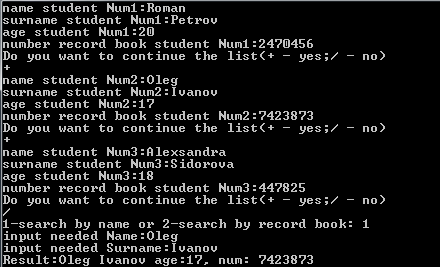
**Рисунок 3 – результат работы программы**

**Задание 4**:



**Рисунок 4 – результат работы программы**

**Задание 5**:



**Рисунок 5 – результат работы программы**

**Вывод**

Заново изучены простые структуры данных, динамическое использование памяти.

**Листинг**

**Задание 1**:

﻿#include <stdio.h>

#include <limits.h>

int main()

{

int max=INT\_MIN, min=INT\_MAX, n = 10, i = 0;

int a[10];

for (i; i < n; i++) {

a[i] = i \* 4 + 3;

printf("%d ", a[i]);

}

i = 0;

for (i; i < n; i++) {

if (a[i] > max) max = a[i];

if (a[i] < min) min = a[i];

}

printf("\n max-min=%d", max-min);

}

**Задание 2**:

﻿#include <stdio.h>

#include <stdlib.h>

int main()

{

int max=100, min=-100, n = 10, i = 0;

int a[10];

for (i; i < n; i++) {

a[i] = min+ rand()%(max-min+1);

printf("a[%d] = %d \ns",i ,a[i]);

}

}

**Задание 3**:

﻿#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

int main()

{

int max=100, min=-100, i = 0, size\_mas=0;

int \*a;

printf("input size mass:");

scanf("%d", &size\_mas);

a = (int\*)malloc(size\_mas \* sizeof(int));

for (i; i < size\_mas; i++) {

a[i] = min+ rand()%(max-min+1);

printf("a[%d] = %d \n",i ,a[i]);

}

free(a);

}

**Задание 4**:

﻿#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

int main()

{

int max=100, min=-100, i = 0,j=0, line=0, column=0, t, sumpr=0;

int \*\*a;

int\* sum=nullptr;

printf("input count line mass:");

scanf("%d", &line);

printf("input count column mass:");

scanf("%d", &column);

a = (int\*\*)malloc((line\*2) \* sizeof(int));

for (i=0; i < line; i++) {

a[i] = (int\*)malloc(column \* sizeof(int));

for (j=0; j < column; j++) {

a[i][j] = min+ rand()%(max-min+1);

printf("%d ",a[i][j]);

}

printf("\n");

}

printf("inut 1 for sum column or input 2 for sum line: ");

scanf("%d", &t);

if (t == 1) {

sum = (int\*)malloc(column \* sizeof(int)+1);

for (j = 0; j < column; j++) {

sumpr = 0;

for (i = 0; i < line; i++) {

sumpr = sumpr + a[i][j];

}

sum[j] = sumpr;

}

for (int k = 0; k < column; k++) {

printf("sum Column %d=%d ", k, sum[k]);

}

free(sum);

}

else if (t == 2) {

sum = (int\*)malloc(line \* sizeof(int) + 1);

for (i = 0; i < line; i++) {

sumpr = 0;

for (j = 0; j < column; j++) {

sumpr = sumpr + a[i][j];

}

sum[i] = sumpr;

}

for (int k = 0; k < column; k++) {

printf("sum line %d=%d ", k, sum[k]);

}

free(sum);

}

else

{

printf("error");

}

for (i = 0; i < line; i++) {

free(a[i]);

}

free(a);

return 0;

}

**Задание 5**:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct student {

char Name[10];

char Famil[15];

char age[10], Nomzach[15];

struct student\* next;

}Node;

int i = 1;

char specchar = '/';

Node\* getLast(Node\* head) {

if (head == NULL) {

return NULL;

}

while (head->next) {

head = head->next;

}

return head;

}

Node\* input\_head() {

Node\* list = (Node\*)malloc(sizeof(Node));

printf("name student Num%d:", i);

scanf("%s", list->Name);

printf("surname student Num%d:", i);

scanf("%s", list->Famil);

printf("age student Num%d:", i);

scanf("%s", list->age);

printf("number record book student Num%d:", i);

scanf("%s", list->Nomzach);

list->next = NULL;

return(list);

}

int input(Node\* head) {

int f = 0;

Node\* last = getLast(head);

Node\* list = (Node\*)malloc(sizeof(Node));

printf("name student Num%d:", i);

scanf("%s", list->Name);

printf("surname student Num%d:", i);

scanf("%s", list->Famil);

printf("age student Num%d:", i);

scanf("%s", list->age);

printf("number record book student Num%d:", i);

scanf("%s", list->Nomzach);

list->next = NULL;

last->next = list;

printf("Do you want to continue the list(+ - yes;/ - no)\n");

scanf(" %c", &specchar);

getchar();

if (specchar == '+') {

f = 1;

}

return(f);

}

void searchNF(Node\* head, const char\* Name, const char\* Famil) {

int s = 0;

while (head != NULL) {

if (strcmp(head->Name, Name) == 0 && (strcmp(head->Famil, Famil) == 0)) {

printf("Result:%s %s age:%s, num: %s", head->Name, head->Famil, head->age, head->Nomzach);

s = 1;

break;

}

else

{

head = head->next;

}

}

if (s != 1) {

printf("Not found");

}

}

void searchZ(Node\* head, const char\* Nomzach) {

int s = 0;

while (head != NULL) {

if ((strcmp(head->Nomzach,Nomzach)==0)) {

printf("Result:%s %s age:%s, num: %s", head->Name, head->Famil, head->age, head->Nomzach);

s = 1;

break;

}

else

{

head = head->next;

}

}

if (s != 1) {

printf("Not found");

}

}

int main()

{

char SName[10], SFamil[15], sNomzach[15];

int t=0;

int f = 0;

Node\* head = input\_head();

printf("Do you want to continue the list(+ - yes;/ - no)\n");

scanf(" %c", &specchar);

getchar();

if (specchar == '+') {

f = 1;

}

while (f == 1)

{

i++;

f = input(head);

}

printf("1-search by name or 2-search by record book: ");

scanf("%d", &t);

if (t == 1) {

printf("input needed Name:");

scanf("%s", SName);

printf("input needed Surname:");

scanf("%s", SFamil);

searchNF(head, SName, SFamil);

}

else if (t == 2) {

printf("input needed number record book:");

scanf("%s", sNomzach);

searchZ(head, sNomzach);

}

else

{

printf("Error");

}

return 0;

}