Отчёт по заданию №8

по курсу «Языки и методы программирования».

Выполнил студент группы М8О-111Б-23: Воробьев Глеб Янович № по списку 5.

Контакты: koshastet13@gmail.com

Работа выполнена: «13» апреля 2024 г.

Преподаватель: каф. 806 Никулин Сергей Петрович

Входной контроль знаний с оценкой: _____

Отчет сдан «14» апреля 2024 г.

Итоговая оценка: _____

Подпись преподавателя: _____

1. **Тема:**
Реализация линейных списков на си.
2. Цель работы:
Составить и отладить программу на языке Си для обработки линейного списка заданной организации с отображением списка на динамические структуры
3. Задание:
Реализация линейного однонаправленного списка с барьерным элементом с элементами строкового тип. Предусмотреть выполнение одного нестандартного (удалить каждый k-ый элемент списка) и четырех стандартных действий: печать списка, вставка нового элемента в список, удаление элемента из списка, подсчет длины списка.
4. Оборудование:
Процессор AMD Ryzen 5 7640HS.
ОП 16 ГБ.
SSD 512 ГБ.
Монитор 2560x1600~165Hz.
5. Программное обеспечение:
Операционная система семейства Unix.
Наименование Ubuntu версия 22.04.3.
Интерпретатор команд GNU bash версия 6.2.0.
Система программирования -.
Редактор текстов Visual Studio Code.
6. Идея, метод, алгоритм решения задачи:
Описание работы скрипта:

1) Инициализация: Скрипт начинается с объявления переменной state и, возможно, инициализации указателя head и барьерного элемента barrier. Они используются для управления связанным списком.
2) Ввод состояния: Программа входит в цикл, где она считывает целочисленное значение в переменную state через стандартный ввод.
3) Обработка состояния: В зависимости от значения state, выполняется одна из функций (stateOne, stateTwo, stateThree, stateFour, stateFive), которые выполняют операции над связанным списком.
4) Завершение: Если вводится 0, цикл прерывается, что означает выход из программы.
5) Очистка памяти: Программа вызывает функцию deleteList, чтобы удалить все элементы связанного списка и освободить память, выделенную под барьерный элемент.

7. Сценарий выполнения работы:

1) Заголовочный файл item.h

```c
#ifndef __ITEM_H__
#define __ITEM_H__
#define MAX_LEN 100

typedef struct Item {
   struct Item *next;
   char key[MAX_LEN];
```

```
}Item;

Item *intitBarrier(void);
void stateOne(Item *head, Item *barrier);
void stateTwo(Item **head, Item *barrier);
void stateThree(Item **head, Item *barrier);
void stateFour(Item *head, Item *barrier);
void stateFive(Item **head, Item *barrier);
Item *createNode(Item *barrier, char str[]);
void addNode(Item **head, Item *barrier, int pos, char add[]);
void printList(Item *head, Item *barrier);
void deleteNode(Item **head, Item *barrier, char str[]);
void deleteList(Item **head, Item *barrier);
int sizeList(Item *head, Item *barrier);
void specialAct(Item **head, Item *barrier, int k);

#endif
```

2) Основной скрипт main.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "item.h"
#define MAX_LEN 100

int main()
{
   char headValue[MAX_LEN];
   int state;
   printf("Put first element of list\n");

   Item *barrier = intitBarrier();

   scanf("%s", headValue);
   Item *head = createNode(barrier, headValue);

   printf("Choose option: 1 - print tree, 2 - add new element, 3 - delete
element, 4 -  size of list, 5 - delete every k-th element, 0 - finish
script\n");

   while (scanf("%d", &state))
   {
       if (state == 1) {
           stateOne(head, barrier);
       } else if (state == 2) {
           stateTwo(&head, barrier);
       } else if (state == 3) {
```

```
            stateThree(&head, barrier);
        } else if (state == 4) {
            stateFour(head, barrier);
        } else if (state == 5) {
            stateFive(&head, barrier);
        } else if (state == 0) {
            break;
        } else {
            printf("Choose option from 0 to 5\n");
        }
    }

    deleteList(&head, barrier);
    free(barrier);

    return 0;
}
Item *intitBarrier(void)
{
    Item *barrier = malloc(sizeof(Item));

    barrier->next = barrier;

    return barrier;
}

void stateOne(Item *head, Item *barrier)
{
    printList(head, barrier);

    printf("Chose another option\n");
}

void stateTwo(Item **head, Item *barrier)
{
    char add[MAX_LEN];
    int pos;

    printf("Give positon and value:\n");

    scanf("%d %s", &pos, add);

    addNode(head, barrier, pos, add);

    printf("Chose another option\n");
}

void stateThree(Item **head, Item *barrier)
```

```
{
    char del[MAX_LEN];

    printf("Choose node to delete:\n");
    scanf("%s", del);

    deleteNode(head, barrier, del);
    printf("Chose another option\n");
}

void stateFour(Item *head, Item *barrier)
{
    int size = sizeList(head, barrier);

    printf("Size of list is %d\n", size);
    printf("Chose another option\n");
}

void stateFive(Item **head, Item *barrier)
{
    int k;

    printf("Choose k:\n");
    scanf("%d", &k);

    specialAct(head, barrier, k);

    printf("Chose another option\n");
}

Item *createNode(Item *barrier, char str[])
{
    Item *tmp = malloc(sizeof(Item));

    strncpy(tmp->key, str, MAX_LEN);
    tmp->key[MAX_LEN] = '\0';

    tmp->next = barrier;

    return tmp;
}

void addNode(Item **head, Item *barrier, int pos, char add[])
{
    int size = sizeList(*head, barrier);

    Item *newNode = createNode(barrier, add);
```

```
    if (pos < 0 || pos > size) {
        printf("Invalid position to insert\n");

    } else if (pos == 0) {
        newNode->next = *head;
        *head = newNode;

    } else {
        Item *tmp = *head;

        while (--pos) {
            tmp = tmp->next;
        }

        newNode->next = tmp->next;
        tmp->next = newNode;
    }
}

void printList(Item *head, Item *barrier)
{
    printf("Your list:\n");

    while(head != barrier) {
        printf("%s ", head->key);
        head = head->next;
    }

    printf("\n");
}

void deleteNode(Item **head, Item *barrier, char str[])
{
    Item *tmp = *head;
    Item *prv;

    if (tmp->next == barrier) {
        *head = barrier;
        free(tmp);
        printf("Value: %s, deleted\n", str);
        return;
    }

    if (tmp != barrier && strcmp(str, tmp->key) == 0) {
        *head = tmp->next;
        printf("Value: %s, deleted\n", str);
        free(tmp);
        return;
```

```
    }

    while(tmp != barrier && strcmp(str, tmp->key) != 0) {
        prv = tmp;
        tmp = tmp->next;
    }

    if (tmp == barrier) {
        printf("Value not founded\n");
        return;
    }

    prv->next = tmp->next;
    free(tmp);

    printf("Value: %s, deleted\n", str);
}

void deleteList(Item **head, Item *barrier)
{
    Item *tmp = *head;
    Item *next;

    while (tmp != barrier) {
        next = tmp->next;
        free(tmp);
        tmp = next;
    }

    *head = barrier;
}

int sizeList(Item *head, Item *barrier)
{
    int size = 0;

    while (head != barrier) {
        size++;
        head = head->next;
    }

    return size;
}

void specialAct(Item **head, Item *barrier, int k)
{
    Item *tmp = *head;
    Item *prv;
```

```
    int count = 0;

    if (tmp == barrier || k <= 0) {
        printf("Error. Choose anotherl k\n");
        return;
    }

    if (k == 1) {
        deleteList(head, barrier);
        printf("List was deleted\n");
        return;
    }

    while(tmp != barrier) {
        count++;

        if (k == count) {
            prv->next = tmp->next;
            free(tmp);
            tmp = prv->next;

            count = 0;
        }

        if (count != 0)
            prv = tmp;

        tmp = prv->next;
    }

    printf("Every %d-th element was deleted\n", k);
}
```

Допущен к выполнению работы. **Подпись преподавателя** _____

8. **Распечатка протокола:**

Put first element of list
first
Choose option: 1 - print tree, 2 - add new element, 3 - delete element, 4 - size of list, 5 - delete every k-th element, 0 - finish script
1
Your list:
first
Chose another option
4
Size of list is 1
Chose another option
2
Give positon and value:
1 632
Chose another option
1

Your list:
first 632
Chose another option
4
Size of list is 2
Chose another option
2
Give positon and value:
0 u
Chose another option
1
Your list:
u first 632
Chose another option
4
Size of list is 3
Chose another option
2
Give positon and value:
1 hello
Chose another option
1
Your list:
u hello first 632
Chose another option
4
Size of list is 4
Chose another option
2
Give positon and value:
2 4
Chose another option
1
Your list:
u hello 4 first 632
Chose another option
4
Size of list is 5
Chose another option
3
Choose node to delete:
4
Value: 4, deleted
Chose another option
1
Your list:
u hello first 632
Chose another option
4
Size of list is 4
Chose another option
5
Choose k:
2
Every 2-th element was deleted
Chose another option
1
Your list:
u first
Chose another option
4
Size of list is 2
Chose another option
2
Give positon and value:
2 4
Chose another option
1
Your list:
u first 4
Chose another option
2
Give positon and value:
3 t

```
Chose another option
1
Your list:
u first 4 t
Chose another option
2
Give positon and value:
0 6
Chose another option
1
Your list:
6 u first 4 t
Chose another option
4
Size of list is 5
Chose another option
2
Give positon and value:
1 7
Chose another option
1
Your list:
6 7 u first 4 t
Chose another option
2
Give positon and value:
3 io
Chose another option
1
Your list:
6 7 u io first 4 t
Chose another option
4
Size of list is 7
Chose another option
5
Choose k:
3
Every 3-th element was deleted
Chose another option
1
Your list:
6 7 io first t
Chose another option
4
Size of list is 5
Chose another option
3
Choose node to delete:
t
Value: t, deleted
Chose another option
1
Your list:
6 7 io first
Chose another option
5
Choose k:
2
Every 2-th element was deleted
Chose another option
1
Your list:
6 io
Chose another option
5
Choose k:
1
List was deleted
Chose another option
1
Your list:

Chose another option
```

```
2
Give positon and value:
0 9
Chose another option
1
Your list:
9
Chose another option
4
Size of list is 1
Chose another option
3
Choose node to delete:
4
Value: 4, deleted
Chose another option
1
Your list:

Chose another option
4
Size of list is 0
Chose another option
0
[1] + Done                    "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm}
0<"/tmp/Microsoft-MIEngine-In-xsd33rjw.4uc" 1>"/tmp/Microsoft-MIEngine-Out-oubvyapm.udl"
```

9. Дневник отладки

№	Лаб. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. Замечания автора:

По существу работы: замечания отсутствуют.

11. Выводы:

В ходе выполнения данного задания практикума я научился работать с линейными списками в СП С.

Подпись студента _____