

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных
систем

Лабораторная работа №2

по дисциплине: Алгоритмы и структуры данных

тема: «Производные структуры данных.

Структура данных типа «строка»»

Выполнил: ст. группы ПВ-202
Буйвало Анастасия Андреевна

Проверил:
Кабалянц Петр Степанович
Маньшин Илья Михайлович

Белгород 2021 г.

Лабораторная работа №2

«Производные структуры данных. Структура данных типа «строка»»

Цель работы: изучение встроенной структуры данных типа «строка», разработка и использование производных структур данных строкового типа.

Вариант №7:

Номер формата	Задача
7	7

Задание:

1. Для СД типа строка определить:
 - 1.1. Абстрактный уровень представления СД:
 - 1.1.1 Характер организованности и изменчивости.
 - 1.1.2. Набор допустимых операций.
 - 1.2. Физический уровень представления СД:
 - 1.2.1. Схему хранения.
 - 1.2.2. Объем памяти, занимаемый экземпляром СД.
 - 1.2.3. Формат внутреннего представления СД и способ его интерпретации.
 - 1.2.4. Характеристику допустимых значений.
 - 1.2.5. Тип доступа к элементам.
 - 1.3. Логический уровень представления СД.
Способ описания СД и экземпляра СД на языке программирования.
2. Реализовать СД строкового типа в соответствии с вариантом индивидуального задания (см. табл.8) в виде модуля. Определить и обработать исключительные ситуации.
3. Разработать программу для решения задачи в соответствии с вариантом индивидуального задания (см. табл.8) с использованием модуля, полученного в результате выполнения пункта 2.

Заголовок: *function SudWord(s:string;n:word):string/ string1*

**SudWord(char *s, unsigned n).*

Назначение: выделение из строки *s* слов, начиная с номера *n*.

Входные параметры: *s, n*.

Выходные параметры: нет.

Выполнение работы:

Задание 1:

Тип данных строка:

Абстрактный уровень представления СД:

1. Характер организованности – простейшие;
2. Изменчивость – динамическая СД;
3. Набор допустимых операций: доступ, присваивание, инициализация

Физический уровень представления СД:

1. Схема хранения: последовательная.
2. Объем памяти, занимаемый экземпляром СД: $1+N$ (количество символов в строке) байт.
3. Формат внутреннего представления СД и способ его интерпретации:

s[0]	...	s[N-1]	"\0"
1 + N байт			

s[0] – первый символ в строке

s[N-1]-последний символ в строке

N – количество символов в строке (символы из таблицы ASCII)

4. Характеристику допустимых значений.
от 0 до 255 для каждого элемента строки
5. Тип доступа к элементам: прямой

Логический уровень представления СД:

Способы описания СД:

char s[N];

typedef char t_s[N];

typedef char *t_s;

Способы описания экземпляра:

t_s str;

Задание 2:

Реализовать СД строкового типа в соответствии с вариантом индивидуального задания (см. табл.8) в виде модуля. Определить и обработать исключительные ситуации.

(__FORM7_H.c)

```
#include <stdio.h>
#include <mm_malloc.h>

#include "__FORM7_H.h"

//Коды ошибок

const int everything_fine = 0;
const int capacity_overflow = 1;
const int input_overflow = 2;
const int out_of_range = 3;

//Переменная ошибок

int StrError = everything_fine;

/*Выделение динамической памяти под строку s в структуре st, содержащую от 0
до n символов.
Значением n определяется максимальное количество символов, которое может
вместить строка
(зависит от кол-ва выделенной памяти).
Динамическая длина строки есть ее текущая длина.*/
void InitStr(string1 st, unsigned n)
{
    if(n > st->max) {
        printf("out_of_range");
        StrError = out_of_range;
    }
    st->s = calloc(n, sizeof(char));
}

/*Ввод строки s структуры st с клавиатуры*/
void InputStr(string1 st)
{
    if(gets(st->s) == USER_ADDR_NULL) {
        printf("input_overflow");
        StrError = input_overflow;
    }
}

/*Вывод строки s структуры st на экран монитора*/
void OutputStr(string1 st)
{
    if(puts(st->s) < 0)
        printf("out error");
}

/*Запись данных в строку s структуры st из строки s2. Строка s2 заканчивается
нулевым символом '\0'*/
void WriteToStr(string1 st, char *s2)
{
    int i = 0;
```

```

        while(s2[i] != '\0'){
            (st->s)[i] = s2[i];
            i++;
        }
        st->s[i] = '\0';
    }

```

/*Запись данных в строку s2 из строки s структуры st. Строка s2 заканчивается нулевым символом '\0'*/

```

void WriteFromStr(char *s2, string1 st)
{
    int i = 0;
    while((st->s)[i] != '\0'){
        s2[i] = (st->s)[i] ;
        i++;
    }
    s2[i] = '\0';
}

```

/* возвращает длину строки s структуры st */

```

unsigned Length(string1 s1)
{
    int i = 0;
    while((s1->s)[i] != '\0' && (s1->s)[i] != '\n')
        i++;
    return i;
}

```

/*Сравнивает строки s структуры s1 и s структуры s2. Возвращает 0 если s структуры s1 = s структуры s2; 1, если s структуры s1 > s структуры s2; -1, если s структуры s1 < s структуры s2.*/

```

int Comp(string1 s1, string1 s2)
{
    int n1 = Length(s1);
    int n2 = Length(s2);
    if(n1 == n2)
        return 0;
    if(n1 > n2)
        return 1;
    else
        return -1;
}

```

/*Записывает Count символов в строку s структуры Subs из строки s структуры s1, начиная с позиции Index */

```

void Copy(string1 s1, unsigned Index, unsigned Count, string1 Subs)
{
    int i = 0;
    for(; i < Count; i++)
        (Subs->s)[i] = (s1->s)[Index + i];
    (Subs->s)[i] = '\0';
}

```

/*Удаляет Count символов из строки s структуры s1, начиная с позиции Index.*/

```

void Delete(string1 s1, unsigned Index, unsigned Count)
{
    if(Index + Count > Length(s1)){
        printf("out_of_range");
        StrError = out_of_range;
    }
    else {
        int i = 0;

```

```

        for(; i < Length(s1) - Count; i++)
            (s1->s)[Index + i] = (s1->s)[Index + i + Count];
        (s1->s)[i] = '\0';
    }
}

```

/*Вставляет подстроку s структуры SubS в строку s структуры s1, начиная с позиции Index. */

```
void Insert(string1 Subs, string1 s1, unsigned Index)
```

```

{
    int k2 = Length(Subs);
    if(Index + k2 > s1->max){
        printf("out_of_range");
        StrError = out_of_range;
    }
    else{
        int k1 = Length(s1);
        int i;
        for(i = Index; i < k1; i++) {
            (Subs->s)[k2++] = (s1->s)[i];
        }
        i = 0;
        for( ; i <= k2; i++)
            (s1->s)[Index++] = (Subs->s)[i];
        (s1->s)[++i] = '\0';
    }
}

```

/*Выполняет конкатенацию строк s структуры s1 и s структуры s2. Результат помещает в s структуры srez*/

```
void Concat(string1 s1, string1 s2, string1 srez)
```

```

{
    int k1 = Length(s1);
    int k2 = Length(s2);
    if(k1 + k2 > srez->max) {
        printf("capacity_overflow");
        StrError = capacity_overflow;
    }
    else {
        int i = 0;
        for (; i < k1; i++)
            (srez->s)[i] = (s1->s)[i];
        int j = 0;
        for (; j < k2; j++)
            (srez->s)[i++] = (s2->s)[j];
        (srez->s)[i] = '\0';
    }
}

```

/*Возвращает позицию, начиная с которой в строке s структуры s2 располагается подстрока s структуры SubS, иначе -1 */

```
unsigned Pos(string1 SubS, string1 s2)
```

```

{
    for(int i = 0; (s2->s)[i] != '\0'; i++){
        int k = 0;
        for(int j = i; (s2->s)[j] == (SubS->s)[k] && k < Length(SubS); j++){
            k++;
        }
        if ((SubS->s)[k] == '\0')
            return i;
    }
}

```

```

        return -1;
    }

    /*Удаляет строку s структуры s1 и структуру s1 из динамической памяти*/
    void DoneStr(string1 s1)
    {
        free(s1->s);
        free(s1);
    }

    (__FORM7_H.h)

    //
    // Created by настя буйвало on 25/09/2021.
    //

    #ifndef INC_2AISD__FORM7_H_H
    #define INC_2AISD__FORM7_H_H

    extern const int everything_fine;
    extern const int capacity_overflow;
    extern const int input_overflow;
    extern const int out_of_range; // Определение исключительных ситуаций

    typedef struct str{
        char *s;
        unsigned max; /* Максимальное количество символов в строке,
определяющееся при инициализации */
    };
    typedef struct str *string1;
    /*Выделение динамической памяти под строку st, содержащую от 0 до n символов.
    Значением n определяется максимальное количество символов, которое может
    вместить строка
    (зависит от кол-ва выделенной памяти).
    Динамическая длина строки есть ее текущая длина.*/
    void InitStr(string1 st, unsigned n);

    /*Ввод строки s с клавиатуры*/
    void InputStr(string1 st);

    /*Вывод строки s на экран монитора*/
    void OutputStr(string1 st);

    void WriteToStr(string1 st, char *s);

    void WriteFromStr(char *s, string1 st);
    void InputStr(string1 st);
    void OutputStr(string1 st);
    int Comp(string1 s1, string1 s2);
    void Delete(string1 s, unsigned Index, unsigned Count);
    void Insert(string1 Subs, string1 s, unsigned Index);
    void Concat(string1 s1, string1 s2, string1 srez);
    void Copy(string1 s, unsigned Index, unsigned Count, string1 Subs);
    unsigned Length(string1 s);
    unsigned Pos(string1 SubS, string1 s);
    void DoneStr(string1 s);
    int StrError; // Переменная ошибок//

```

```
#endif //INC_2AISD___FORM7_H_H
```

(main.c)

```
int main()
{
    string1 s2;
    s2 = calloc(1, sizeof(struct str));
    s2->max = 257;
    InitStr(s2, 256);
    InputStr(s2);

    //OutputStr(s2);

    string1 s1;
    s1 = calloc(1, sizeof(struct str));
    s1->max = 257;
    InitStr(s1, 256);
    InputStr(s1);

    string1 srez;
    srez = calloc(1, sizeof(struct str));
    srez->max = 257;
    InitStr(srez, 256);

    //SudWord(s1, s2, 6);
    //OutputStr(s2);

    printf("Comp(s1, s2) = %d\n", Comp(s1, s2));

    Insert( s1, s2, 2);
    printf("Insert(s1, s2, 2) :");
    OutputStr(s2);

    Delete(s2, 0, 2);
    printf("Delete(s2, 0, 2) :");
    OutputStr(s2);

    Concat(s1, s2, srez);
    printf("Concat(s1, s2, srez) :");
    OutputStr(srez);

    printf("Length(s2) = %d\n", Length(s2));

    printf("Pos(s2, srez) = %d\n", Pos(s2, srez));

    Copy(s1, 1, 2, srez);
    printf("Copy(s1, 1, 2, srez) : ");
    OutputStr(srez);

    DoneStr(s1);
    DoneStr(s2);
    DoneStr(srez);

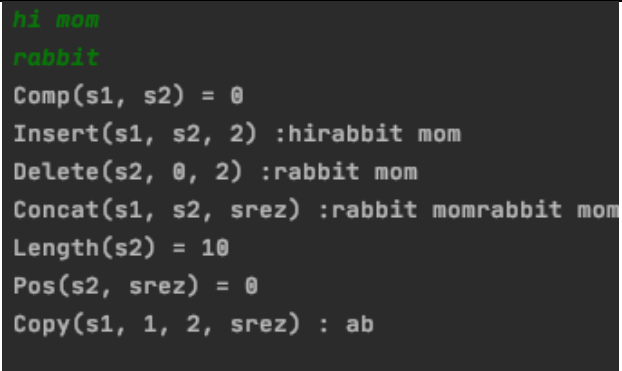
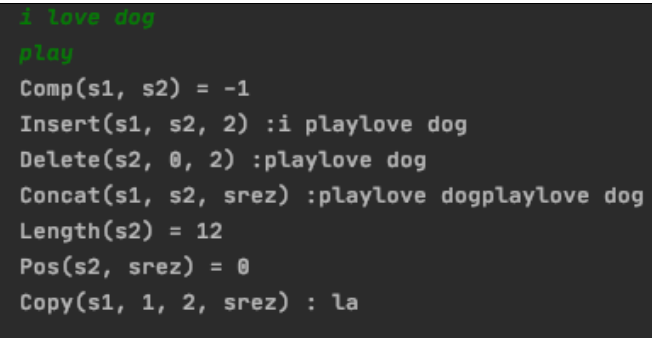
    return 0;
}
```


Задание 3:

Разработать программу для решения задачи в соответствии с вариантом индивидуального задания (см. табл.8) с использованием модуля, полученного в результате выполнения пункта 2.

```
/* выделение из строки s структуры s1 слов и запись их в строку s структуры s2,
начиная с номера n */
```

```
void SudWord(string1 s1, string1 s2, int n)
{
    string1 st;
    st = calloc(1, sizeof(struct str));
    st->max = 257;
    InitStr(st, 256);
    Copy(s1, 0, Length(s1), st);
    Delete(st, 0, n);
    Copy(st, 0, Length(st), s2);
}
```

№	Входные данные (s2, s1)	Результат	
1	S2 = hi mom S1 = rabbit	Comp(s1, s2) :0 Insert(s1, s2, 2) : hirabbit mom Delete(s2, 0, 2) :rabbit mom Concat(s1, s2, srez) : rabbit momrabbit mom Length(s2): 10 Pos(s2, srez) : 0 Copy(s1, 1, 2, srez) : ab	 A screenshot of a terminal window showing the output of the program for the first test case. The output includes the initial comparison (0), insertion of 'hi' at index 2, deletion of the first two characters, concatenation of the original string and the modified string, and the final string 'rabbit momrabbit mom' with its length (10) and position (0). The substring 'ab' is also shown.
2	S2 = i love dog S1 = play	Comp(s1, s2) :-1 Insert(s1, s2, 2): i playlove dog Delete(s2, 0, 2) :playlove dog Concat(s1, s2) :play love dogplay love dog Length(s2):12 Pos(s2, srez) :0 Copy(s1, 1, 2, srez) :la	 A screenshot of a terminal window showing the output of the program for the second test case. The output includes the initial comparison (-1), insertion of 'i' at index 2, deletion of the first two characters, concatenation of the original string and the modified string, and the final string 'playlove dogplay love dog' with its length (12) and position (0). The substring 'la' is also shown.

3	S1 = flood in	SudWord(s1, s2 , 6): in		flood in in	
---	------------------	----------------------------	--	----------------	--

Вывод: в ходе выполнения лабораторной работы, были реализованы функции работы со строками, из которых была составлена библиотека, согласно формату, с помощью этой библиотеки было выполнено задание варианта. Результаты работы программы совпали с предполагаемыми.