**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

Лабораторная работа №1

По дисциплине «Программирование»

 Указатели.

Группа: АВТ-041

Студент: Дударь М.А.

Преподаватель: Балакин В.В.

НОВОСИБИРСК 2021

1. **Постановка задачи:**

Необходимо реализовать функцию, которая находит в строке самое первое (по алфавиту) слово. С ее помощью реализовать размещение слов в выходной строке в алфавитном порядке.

1. **Алгоритм программы:**
2. Считывание строку из консоли.
3. Поиск первые буквы в словах.
4. Сортировка строку по алфавиту.

**3. Методы решения:**

1. Написать функцию считывания строки с консоли.
2. Написать функцию поиска слов в строке и записывать первую букву, ее десятичное представление и длину слова в двумерный динамический массив.
3. Написать функцию сортировки строки по алфавиту, опираясь на десятичное представления первых букв слов.
4. **Принцип работы.**

Функция считывания с консоли строки, создания динамического массива и возвращение указателя на его начало.

char \*readln(int \*counter, int \*size) {  
 char c = ' ';  
 char \*str;  
 str = (char \*) malloc(10 \* sizeof(char));  
 for (int i = 0; c != '\n'; i++) {  
 str = (char \*) realloc(str, (i + 2) \* sizeof(char));  
 scanf("%c", &c);  
 (\*size)++;  
 if (c != '\n') {  
 str[i] = c;  
 str[i + 1] = '\0';  
 if (c == ' ')(\*counter)++;  
 } else { str[i] = '\0'; }  
 }  
 return str;  
}

Функция расширения динамического двумерного массива, который хранит индекс первых букв, их код в десятичной системе и размер слова.

int \*\*alloc(const int \*size) {  
 int \*\*num;  
 num = (int \*\*) malloc(3 \* sizeof(int \*));  
 num[0] = (int \*) malloc(sizeof(int));  
 num[1] = (int \*) malloc(sizeof(int));  
 num[2] = (int \*) malloc(sizeof(int));  
 num[0] = (int \*) realloc(num[0], (\*size + 1) \* sizeof(int));  
 num[1] = (int \*) realloc(num[1], (\*size + 1) \* sizeof(int));  
 num[2] = (int \*) realloc(num[2], (\*size + 1) \* sizeof(int));  
 return num;  
}

Функция сортировки двумерного массива по его второй строке пузырьком. Применяется для перестановки слов в алфавитном порядке по коду первых символов.

void bubbleSort(int \*\*num, int size, char \*str,const int \*arr\_size) {  
 int temp1, temp2, len;  
 char buf[\*arr\_size];  
 for (int i = 0; i < size - 1; i++) {  
 for (int j = (size - 1); j > i; j--) {  
 if (num[1][j - 1] > num[1][j]) {  
 temp1 = num[1][j - 1];  
 temp2 = num[2][j - 1];  
 len = num[0][j] - num[0][j - 1] + num[2][j];  
 for (int m = 0; m < len; m++) {  
 buf[m] = ' ';  
 }  
 buf[len] = '\0';  
// //IT works)  
 for (int k = 0, l = 0; k < len; k++) {  
 if (k <= num[2][j - 1] - 1) {  
 buf[len + k - num[2][j - 1]] = str[num[0][j - 1] + k];  
 }  
 if (k + num[0][j - 1] >= num[0][j]) {  
 buf[l++] = str[k + num[0][j - 1]];  
 }  
 }  
 for (int k = num[0][j - 1]; k < len + num[0][j - 1]; k++) {  
 str[k] = buf[k - num[0][j - 1]];  
 }  
 num[0][j] = num[0][j - 1] + len - num[2][j - 1];  
 num[1][j - 1] = num[1][j];  
 num[2][j - 1] = num[2][j];  
 num[1][j] = temp1;  
 num[2][j] = temp2;  
 }  
 }  
 }  
}

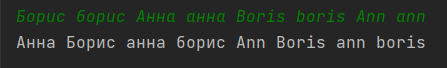
Функция поиска слов в строке и записи в двумерный динамический массив индексов первый букв, их десятичного представления и длины слов.

void counter(char \*str, int \*\*num, int \*arr\_size) {  
 int n = 0;  
 for (int i = 0; i < strlen(str) - 1; i++) {  
 if ((str[i - 1] == ' ') || (i == 0)) {  
 if (str[i] == ' ') continue;  
 num[0][n] = i;  
 num[1][n] = (int) str[i];  
 n++;  
 }  
 }  
 int j = 0, l = 0;  
 for (int i = 0; i <= strlen(str); i++) {  
 j++;  
 if (str[i] == ' ' || str[i] == '\0') {  
 num[2][l] = j - 1;  
 l++;  
 j = 0;  
 }  
 }  
 bubbleSort(num, n, str, arr\_size);  
}

**6.Вывод:**

В ходе выполнения задания была произведена работа с указателями на языке C. Реализованы указатели, взаимодействия с ними, создание и управние динамическими массивами.

Приложение 1. Вывод программы.



Приложение 2. Текст программы.

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
  
char \*readln(int \*counter, int \*size) {  
 char c = ' ';  
 char \*str = (char \*) malloc(10 \* sizeof(char));  
 for (int i = 0; c != '\n'; i++) {  
 str = (char \*) realloc(str, (i + 2) \* sizeof(char));  
 scanf("%c", &c);  
 (\*size)++;  
 if (c != '\n') {  
 str[i] = c;  
 str[i + 1] = '\0';  
 if (c == ' ')(\*counter)++;  
 } else { str[i] = '\0'; }  
 }  
 return str;  
}  
  
  
void dynamic\_array\_print(int \*\*matrix, size\_t N, size\_t M) {  
 for (int i = 0; i < N; i++) {  
 for (int j = 0; j < M; j++) {  
 printf("%\*d", 5, matrix[i][j]);  
 }  
 printf("\n");  
 }  
}  
  
void bubbleSort(int \*\*num, int size, char \*str,const int \*arr\_size) {  
 int temp1, temp2, len;  
 char buf[\*arr\_size];  
 for (int i = 0; i < size - 1; i++) {  
 for (int j = (size - 1); j > i; j--) {  
 if (num[1][j - 1] > num[1][j]) {  
 temp1 = num[1][j - 1];  
 temp2 = num[2][j - 1];  
 len = num[0][j] - num[0][j - 1] + num[2][j];  
 for (int m = 0; m < len; m++) {  
 buf[m] = ' ';  
 }  
 buf[len] = '\0';  
 //IT works)  
 for (int k = 0, l = 0; k < len; k++) {  
 if (k <= num[2][j - 1] - 1) {  
 buf[len + k - num[2][j - 1]] = str[num[0][j - 1] + k];  
 }  
 if (k + num[0][j - 1] >= num[0][j]) {  
 buf[l++] = str[k + num[0][j - 1]];  
 }  
 }  
 for (int k = num[0][j - 1]; k < len + num[0][j - 1]; k++) {  
 str[k] = buf[k - num[0][j - 1]];  
 }  
 num[0][j] = num[0][j - 1] + len - num[2][j - 1];  
 num[1][j - 1] = num[1][j];  
 num[2][j - 1] = num[2][j];  
 num[1][j] = temp1;  
 num[2][j] = temp2;  
 }  
 }  
 }  
}  
  
int \*\*alloc(const int \*size) {  
 int \*\*num;  
 num = (int \*\*) malloc(3 \* sizeof(int \*));  
 num[0] = (int \*) malloc(sizeof(int));  
 num[1] = (int \*) malloc(sizeof(int));  
 num[2] = (int \*) malloc(sizeof(int));  
 num[0] = (int \*) realloc(num[0], (\*size + 1) \* sizeof(int));  
 num[1] = (int \*) realloc(num[1], (\*size + 1) \* sizeof(int));  
 num[2] = (int \*) realloc(num[2], (\*size + 1) \* sizeof(int));  
 return num;  
}  
  
void counter(char \*str, int \*\*num, int \*arr\_size) {  
 int n = 0;  
 for (int i = 0; i < strlen(str) - 1; i++) {  
 if ((str[i - 1] == ' ') || (i == 0)) {  
 if (str[i] == ' ') continue;  
 num[0][n] = i;  
 num[1][n] = (int) str[i];  
 n++;  
 }  
 }  
 int j = 0, l = 0;  
 for (int i = 0; i <= strlen(str); i++) {  
 j++;  
 if (str[i] == ' ' || str[i] == '\0') {  
 num[2][l] = j - 1;  
 l++;  
 j = 0;  
 }  
 }  
 bubbleSort(num, n, str, arr\_size);  
}  
  
int main() {  
 int count = 1, size =0;  
 char \*str = readln(&count,&size);  
 int \*\*num = alloc(&count);  
 counter(str, num,&size);  
 printf("String = %s\n", str);  
 return 0;  
}