МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Программирование»
Тема: “Создание make-файла ”

Студент гр. 7381 _____ Габов Е.С

Преподаватель _____ Берленко Т.А

Санкт-Петербург 2017

Целью данной работы является:
Создать проект, состоящий из пяти файлов: main.c, print_str.c, get_name.c, print_str.h, get_name.h.

- Файл get_name.c должен содержать **описание** функции, которая **считывает** из входного потока имя пользователя и возвращает его.
- Файл get_name.h должен содержать **прототип** функции, которая **считывает** из входного потока имя пользователя и возвращает его.
- Файл print_str.c должен содержать **описание** функции, которая **принимает** в качестве аргумента строку и выводит её (функция ничего не возвращает).
- Файл print_str.h должен содержать **прототип** функции, которая **принимает** в качестве аргумента строку и выводит её (функция ничего не возвращает).
- Файл main.c содержит главную функцию, которая вызывает функцию из файла get_name.h, добавляет к результату выполнения функции строку “Hello, ” и передает полученную строку в функцию вывода строки из print_str.h.

После того, как проект будет готов, создать для него Makefile.

Основные теоретические положения:

В функции main объявляется строковая переменная типа char ‘hello’ и указатель типа char* ‘result’. Далее переменной result присваивается возвращаемое значение функции get_name. Функция get_name описанная в файле get_name.c. В нем сначала динамически выделяется блок памяти с помощью функции malloc при этом указатель переменная name (которая является возвращаемой переменной) указывает на начало выделенного блока памяти. Далее с помощью функции getchar (которая считывает stdin посимвольно) пользователь вводит своё имя. Последнему элементу строки присваивается значение конца строки. Функция возвращает значение name .

В функции main вызывается функция print_str (которая описанная в файле print_str.c) в неё подается функция strncat (функция объединения строк , в данном случае она объядиняет значение переменной hello и result). Функция print_str по условию ничего не возвращает. Она выводит на экран результат работы функции strncat.

В конце функции main очищается место выделенное под значение переменной result при помощи функции free.

В данной работе использовались библиотеки:

1)stdio(библиотеке вводы и вывода)

2)stdlib(библиотека выделения памяти; преобразования типов)

3)string(библиотека работы со строками)

Созданы заголовочные файлы get_name.h и print_str.h. В них используется #pragma once .

Это позволяет быть уверенным в единоразовом подключение заголовочных файлов.

Далее создается Make-файл в котором компилируется программа.

Вывод:

В данной лабораторной работе изучены новые фунции: malloc , strncat , free. Изучены: механизм единоразового подключения заголовочныч файлов с помощью #pragma once, механизм сборки make-файла, использование указателей .

В работе использовались новые такие библиотеки как : stdio.h stdlib.h string.h.

Создана программа, на вход которой подаётся имя пользователя, программа обрабатывает полученный на вход данные и выдает: Hello, <имя пользователя> .

Исходный код программы

Make-файл:

```
all: main.o get_name.o print_str.o
	gcc main.o get_name.o print_str.o
main.o: main.c get_name.h print_str.h
	gcc -c main.c
get_name.o: get_name.c get_name.h
	gcc -c get_name.c
print_str.o: print_str.c print_str.h
	gcc -c print_str.c
```

main.c:

```
#include "get_name.h"
#include "print_str.h"
#include <stdlib.h>
#include <string.h>

int main(){
   char hello[90] = "Hello, ";
   char* result;
   result = get_name();
   print_str(strncat(hello, result, 80));
   free(result);
   return 0;
}
```

Get_name.c:

```
#include "get_name.h"

char* get_name(){
   char* name = (char*)malloc(80*sizeof(char));
   int i = 0;
   char ch;
   while ((ch = getchar()) != '\n')
   {
     name[i] = ch;
     i++;
   }
```

```
    name[i] = '\0';
    return name;
}
```

Print_str:

```
#include "print_str.h"

void print_str(char* str)
{
  puts(str);
}
```