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

ОТЧЕТ
по лабораторной работе No1
по дисциплине «Программирование»
Тема: «Создание make-файла»

Студент гр. 7381 Ильясов А.В.
Преподаватель Берленко Т.А.

Санкт-Петербург
2017

Цель работы.

Изучить основы сборки Makefile (писать цели, инструкции и команды), научиться пользоваться терминалом в ОС Linux, компилировать исходные файлы, создавать заголовочные файлы. Познакомиться с системой контроля версий github.

Создать проект, состоящий из пяти файлов: "get_name.h", "get_name.c", "print_str.h", "print_str.c", "main.c":

Файл "get_name.h" должен содержать прототип функции "char* get_name()", которая считывает из входного потока имя пользователя и возвращает его.

Файл "get_name.c" должен содержать определение функции "char* get_name()", которая считывает из входного потока имя пользователя и возвращает его.

Файл "print_str.h" должен содержать прототип функции "print_str(char*)", которая принимает в качестве аргумента строку и выводит её.

Файл "print_str.c" должен содержать определение функции "print_str(char*)", которая принимает в качестве аргумента строку и выводит её.

Файл "main.c" содержит главную функцию "int main()", которая вызывает "get_name()" из файла "get_name.h", добавляет к результату выполнения функции строку "Hello, " и передаёт полученную строку в качестве аргумента в функцию вывода строки "print_str(char*)" из файла "print_str.h"

После создания проекта, написать Makefile, с помощью которого данный проект будет собираться.

Основные теоретические положения.

Заголовочные файлы стандартной библиотеки языка С, необходимые для выполнения данной лабораторной работы:

1. Функция «void print_str(char* string)». В этой функции заголовочный файл «stdio.h» (стандартный заголовочный файл ввода-вывода) подключает прототип функции «int puts(const char *str)», которая склеивает строку приветствия и имя.

Описание:

Функция «puts» выводит строку типа «char*», на которую указывает параметр «string» в стандартный поток вывод и добавляет символ новой строки '\n'. Функция начинает копировать строку с адреса, указанного в string, пока не достигнет нулевого символа ". Этот заключительный, нулевой символ не копируется в стандартный поток вывод.

Параметры:

«char* string» - С-строка для вывода на стандартный поток вывода.

Возвращаемое значение:

В случае успеха, возвращается неотрицательное значение. В случае ошибки, функция возвращает значение EOF.

2. Функция «int main()». В этой функции заголовочный файл «string.h» (стандартной библиотеки языка Си) подключает прототип функции «char * strncat(char * destptr, char * srcptr, size_t num)», которая склеивает строку приветствия и имя.

Описание:

Функция добавляет первые «num» символов строки «srcptr» к концу строки «destptr», плюс символ конца строки. Если строка «srcptr» больше чем количество копируемых символов «num», то после скопированных символов неявно добавляется символ конца строки.

Параметры:

«destptr» - Указатель на строку назначения, которая будет содержать результат конкатенации строк, включая символ завершения строки.

«srcptr» - Строка, из которой будут копироваться первые «num» символов для конкатенации. «num» - Максимальное количество символов для конкатенации.

Возвращаемое значение:

Указатель на строку с результатом конкатенации.

3. Функция «char* get_name()». В этой функции заголовочный файл «stdlib.h» подключает прототип функции «void *malloc(size_t sizemen)», которая склеивает строку приветствия и имя.

Описание:

Функция «malloc» выделяет блок памяти, размером «sizemem» байт, и возвращает указатель на начало блока. Содержание выделенного блока памяти не инициализируется, оно остается с неопределенными значениями.

Параметры:

«sizemem» - Размер выделяемого блока памяти в байтах.

Возвращаемое значение:

Указатель на выделенный блок памяти. Тип данных на который ссылается указатель всегда «void *», поэтому это тип данных может быть приведен к желаемому типу данных. Если функции не удалось выделить требуемый блок памяти, возвращается нулевой указатель.

4. Функция «int main()». В этой функции заголовочный файл «stdlib.h» подключает прототип функции «void free(void*ptrmem)».

Описание:

Функция «free» освобождает место в памяти. Блок памяти, ранее выделенный с помощью вызова «malloc», «calloc» или «realloc» освобождается. То есть освобожденная память может дальше использоваться программами или ОС. Обратите внимание, что эта функция оставляет значение «ptr» неизменным, следовательно, он по-прежнему указывает на тот же блок памяти, а не на нулевой указатель.

Параметры:

«ptrmem» - Указатель на блок памяти, ранее выделенный функциями «malloc» , «calloc» или «realloc» , которую необходимо высвободить. Если в качестве аргумента передается нулевой указатель, никаких действий не происходит.

Возвращаемое значение:

Функция не имеет возвращаемое значение.

Выводы.

Во время работы над проектом была освоена работа в терминале ОС Linux: создание новых директорий; перемещение между ними; использование текстового редактора; создание новых файлов. Был изучен процесс компилирования исходныъ файлов (gcc), создавание объектных файлов (gcc -o), заголовочных файлов (*.h), «Makefile» (написание целей, зависимостей, команд). Была изучена система контроля версий «github».

Исходный код проекта:

файл main.c

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

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

Файл get_name.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#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;
}
```

Файл print_str.c

```
#include <stdio.h>
#include "print_str.h"

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

Файл get_name.h

```
char* get_name();
```

Файл print_str.h

```
#pragma once void

print_str(char*);
```

Файл Makefile

```
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
```