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

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

Студентка гр. 7381 _____ Алясова А.Н.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург
2017

Цель работы:

Познакомиться с операционной системой Linux. Познакомиться с системой контроля версий git. Научиться создавать проекты из нескольких файлов и писать для них Makefile.

Задание:

Создать проект, состоящий из пяти файлов:
1**. main.c,**
2. **print_str.c**,
3. **get_name.c**,
4. **print_str.h**,
5. **get_name.h**.

Создать для него **Makefile**.

- 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**.

Основные теоретические положения:

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

- ***stdio.h*** – заголовочный файл стандартной библиотеки языка Си, содержащий определения макросов, константы и объявления функций и типов, используемых для различных операций стандартного ввода и вывода
- ***stdlib.h*** - заголовочный файл стандартной библиотеки языка Си, который содержит в себе функции, занимающиеся выделением памяти, контроль процесса выполнения программы, преобразования типов и другие.

- ***string.h*** - заголовочный файл стандартной библиотеки языка Си, содержащий функции для работы с нуль-терминированными строками и различными функциями работы с памятью.

<stdio.h> - содержит прототип функции "void puts(const char* string)", выводящей в поток вывода строку string. Используется в определении функции "print_str(char*)".

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:

"const 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.

<string.h> - содержит прототип функции "char* strncat(char* destptr, char* srcptr, size_t num)", необходимая для склейки приветствия и имени.

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.

<stdlib.h> - содержит функции для выделения и освобождения памяти.
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.

void* malloc(size_t sizemem);

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.

Вывод:

В процессе работы над проектом, были изучены новые функции: malloc, stncat, free. Был изучен механизм единоразового подключения заголовочных файлов с помощью #pragma once. Механизм сборки make-файла и использование указателей были успешно освоены.

В работе использовались заголовочные файлы стандартной библиотеки языка Си такие, как stdio.h stdlib.h string.h .

В результате лабораторной работы была создана программа, на вход которой подается имя пользователя, программа обрабатывает полученные на вход данные и выдает: “ Hello, <имя пользователя> ”.

Исходный код проекта:

*Файл **"main.c"**:*

```
#include "get_name.h"
#include "print_str.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;
}
```

*Файл **Makefile**:*

```
all: main.o get_name.o print_str.o
gcc -o lr1 main.o get_name.o print_str.o

main.o: main.c get_name.h print_name.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
```