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

ОТЧЕТ

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

Студентка гр. 7381 _____ Мартьянова Н. М.

Преподаватель _____ Берленко Т. А.

Санкт-Петербург
2017

Цель работы:

- Освоить ОС Linux;
- Изучить основы сборки Makefile;
- Познакомиться с системой контроля версий git;

Основные теоретические положения

1. Функция **puts**

Прототип функции :

int puts(const char * string);

Заголовочный файл:

stdio.h

Описание:

Функция выводит строку в стандартный поток вывода. Посл е вывода строки производится переход на новую строку. Символ конца строки не выводится.

Параметры:

- **String**

  указатель на строку, которую необходимо вывести.

Возвращаемое значение:

В случае успеха - неотрицательное значение.
В случае ошибки - функция EOF.

2. Функция **strncat**

Прототип функции :

char * strncat(char * destptr, char * srcptr, num);

Заголовочный файл:

string.h

Описание:

Функция добавляет первые num символов строки srcptr к концу строки destptr, пока не встретится символ конца строки. Символ конца строки помещается в конце объединенных строк.

Параметры:

- **destptr**

  указатель на массив в который будет добавлена строка.

- **srcptr**
  Строка, из которой будут копироваться первые num символов.

- **num**
  Максимальная длина добавляемой строки.

Возвращаемое значение:

Функция возвращает указатель на массив, в который добавлена строка.

3. Функция **free**

Прототип функции:

vo id free(void * ptrmem);

Заголовочный файл:

stdlib.h

Описание

Функция free освобождает место в памяти. То есть освобожденная память может дальше использоваться программами или ОС.

Функция оставляет значение ptrmem неизменным.

Параметры:

- **ptrme m**
  Указатель на ранее веделенный блок памяти, который необходимо освободить.

Возвращаемое значение:

Функция не имеет возвращаемого значение.

4. Функция **getchar**

Прототип функции :

int getchar (void);

Заголовочный файл

stdio.h

Описание:

Функция получает символ из стандартного потока ввода.
Параметры:

нет

Возвращаемое значение:

В случае успешного чтения символа возвращается код считанного символа; Если достигнут конец файла, то возвращается EOF.

5. Функция **malloc**

Прототип функции:

void * malloc(size_t sizemem);

Заголовочный файл:

stdlib.h

Описание:

Функция malloc выделяет блок памяти размером sizemem .яя

Параметры:

- **sizeme m**
  Размер выделяемого блока памяти в байтах.

Возвращаемое значение:

В случае успешного резервирования блока памяти функция возвращает указатель на только что выделенный блок.

При неудачном результате операции с памятью функция возвращает NULL.

Вывод:

В ходе работы был освоен терминал ОС Linux, его команды для работы с файлами; был изучен механизм сборки приложения, с помощью make-файла; произошло ознакомление с системой контроля версий git.

Исходный код проекта:

- Файл get_name.c

```
#include <stdio.h>
#include <stdlib.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 get_name.h

```
#pragma once
char* get_name();
```

- Файл print_str.c

```
#include <stdio.h>
#include “print_str(const char* str){
puts(str);
}
```

- Файл print_str.h

```
#pragma once
void print_str(const char*);
```

- Файл main.c

```
#include <string.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;
}
```

- Makefile

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