**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 makefile и работа с git

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Кулигин А.И. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2016

**Цель работы:** Научиться создавать Makefile и ознакомиться с системой контроля версий git.

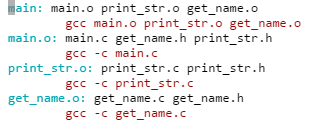
**Ход работы:**

В ветке Kuligin\_lr1 репозитория pr-2016-6303 создали папку kuligin с проектом, состоящим из 5 файлов: main.c, print\_str.h, print\_str.c, get\_name.h, get\_name.c.

1.Мы создали 5 файлов, необходимых для задания: **main.c, print\_str.h**, **print\_str.c**, **get\_name.h** и **get\_name.c** в каталоге kuligin.

* 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**.
  1. **int** main(){
  2. **char** hello[90] = "Hello, ";
  3. **char**\* result;
  4. result = get\_name();
  5. print\_str(strncat(hello, result, 80));
  6. free(result);
  7. **return** 0;
  8. }
* Файл **print\_str.h** должен содержать прототип функции, которая принимает в качестве аргумента строку и выводит её (функция ничего не возвращает).
  1. **void** print\_str(**char**\*);
* Файл **print\_str.c** содержит описание функции, которая принимает в качестве аргумента строку и выводит её (функция ничего не возвращает).
  1. **void** print\_str(**char**\* output) {
  2. puts(output);
  3. }
* Файл **get\_name.h** содержит прототип функции, которая считывает из входного потока имя пользователя и возвращает его.
  1. **char**\* get\_name();
* Файл **get\_name.c** содержит описание функции, которая считывает из входного потока имя пользователя и возвращает его.
  1. **char**\* get\_name(){
  2. **char**\* name = (**char**\*)malloc(80\***sizeof**(**char**));
  3. **int** i = 0;
  4. **char** ch;
  5. **while** ((ch = getchar()) != '\n')
  6. {
  7. name[i] = ch;
  8. i++;
  9. }
  10. name[i] = '\0';
  11. **return** name;
  12. }

2.Создали Makefile со следующим содержанием:



Где до символа “:” идет цель, после “:” идут зависимости, а в новой строке вызываемая функция для получения цели.

Затем загрузили созданную папку с файлами в репозиторий на github с

помощью следующих команд:

**git add kuligin**

**git commit –m nano print\_str.h**

**stud\_40@linux:~/pr1-2016-6303/kuligin$ make**

**gcc -c main.c**

**git push origin Kuligin\_lr1**

**Вывод:** в ходе работы получили навыки работы с системой контроля версий git, а также научился создавать Makefile, упрощающий процесс компиляции.