Оценка 3

1. Используйте git status, чтобы узнать, на какой ветке вы находитесь.

```
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
```

2. Как выглядит git log?

```
commit d44bf5b22122169461bd5ae044382fc64dd5ec4a (HEAD -> main, origin/main, origin/HEAD)
Author: annkits <annpodolskaya0965@gmail.com>
Date:   Sat Mar 8 08:36:27 2025 +0700

    Update and rename lab2.c to linked_lists.c

commit 4cde8f6e21a8a1cd1e544d18c8624a7ebec6530a
Author: annkits <annpodolskaya0965@gmail.com>
Date:   Mon Feb 24 15:41:30 2025 +0700

    Add files via upload

commit 51cd71f6b073505825ff2017bfb58f1b463ec2c8
Author: annkits <annpodolskaya0965@gmail.com>
Date:   Sat Feb 22 12:19:28 2025 +0700
```

3. Создайте файл sort.c и вставьте туда код функции любой сортировки

4. Как сейчас выглядит вывод git status?

```
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        git moves/

nothing added to commit but untracked files present (use "git add" to track)
```

5. Добавьте файл в область stage (add)

```
$ git add .
```

6. Как сейчас выглядит git status?

```
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   git moves/chronology.docx
        new file:   git moves/sort.c
```

7. Закоммитить файл в репозиторий

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/прога/С2-433$ git commit -m "Сообщение коммита:
[main e95a8b6] Сообщение коммита: что поменялось
 2 files changed, 22 insertions(+)
 create mode 100644 git moves/chronology.docx
 create mode 100644 git moves/sort.c
```

8. Как сейчас выглядит git status?

```
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

9. Добавить комментарий с любым текстом в этот же файл

```
 9  //это всё какой-то ужас
10
```

10. Как сейчас выглядит git status?

```
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   git moves/sort.c

no changes added to commit (use "git add" and/or "git commit -a")
```

11. Добавьте (add) изменение файла

12. Как сейчас выглядит git status?

```
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    git moves/chronology.docx -> git_moves/chronology.docx
        renamed:    git moves/sort.c -> git_moves/sort.c
```

13. Измените файл еще раз (можно добавить еще комментарий или убрать старый)

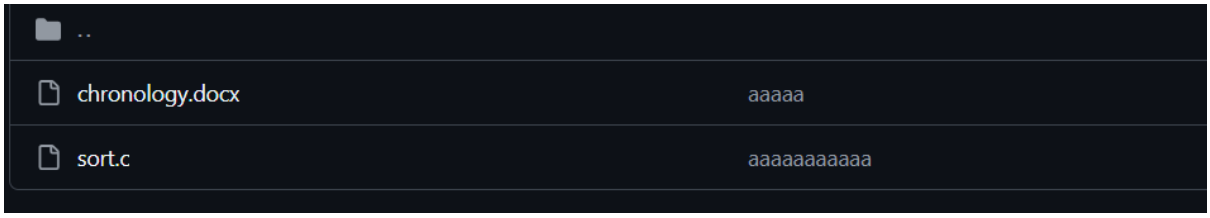
14. Сделайте коммит

15. Как сейчас выглядит status? Журнал (log)?

```
[main 989e637] aaaaaaaaaaa
 1 file changed, 1 insertion(+), 1 deletion(-)
```

```
commit 989e637c6a64e0da821395ffbd95a4bddfa123f6 (HEAD -> main)
Author: annkits <annpodolskaya0965@gmail.com>
Date:   Sat Mar 8 09:26:41 2025 +0700

    aaaaaaaaaaa

commit 82214930e20e317f4c3df90398c5b8881ea3d6ad
Author: annkits <annpodolskaya0965@gmail.com>
Date:   Sat Mar 8 09:18:09 2025 +0700

    aaaaa

commit e95a8b6d12c3a7454dcebcfdc583bc3858131a5c
Author: annkits <annpodolskaya0965@gmail.com>
Date:   Sat Mar 8 09:07:20 2025 +0700
```

16.



Ветки

1. Используйте git branch mybranch (или git checkout -b mybranch), чтобы создать новую ветку с именем mybranch.

```
* main
  mybranch
```

2. Снова используйте git branch, чтобы увидеть новую созданную ветку.

3. Используйте git switch mybranch (или git checkout mybranch), чтобы переключиться на новую ветку.

```
Switched to branch 'mybranch'
```

4. Как изменяется вывод git status при переключении между master и новой веткой, которую вы создали?

```
On branch mybranch
nothing to commit, working tree clean
```

6. Создайте файл с именем file1.txt и своим именем.

7. Добавьте файл и закоммитьте это изменение.

```
[mybranch b0e7d3b] txt file
 1 file changed, 1 insertion(+)
 create mode 100644 git_moves/file1.txt
```

8. Используйте git log --oneline --graph, чтобы увидеть, что ваша ветка указывает на новый коммит.

```
* b0e7d3b (HEAD -> mybranch) txt file
* 989e637 (origin/main, origin/HEAD, main) aaaaaaaaaaa
* 8221493 aaaaa
* e95a8b6 Сообщение коммита: что поменялось
* d44bf5b Update and rename lab2.c to linked_lists.c
* 4cde8f6 Add files via upload
* 51cd71f Add files via upload
```

9. Вернитесь к ветке с именем master.

```
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

10. Используйте git log --oneline --graph, что изменилось? HEAD, txt file commit

```
* 989e637 (HEAD -> main, origin/main, origin/HEAD) aaaaaaaaaaa
* 8221493 aaaaa
* e95a8b6 Сообщение коммита: что поменялось
* d44bf5b Update and rename lab2.c to linked_lists.c
* 4cde8f6 Add files via upload
* 51cd71f Add files via upload
```

11. Создайте новый файл с именем file2.txt и закоммитьте его.

```
[main 2b47d27] file2.txt
 1 file changed, 1 insertion(+)
 create mode 100644 file2.txt
```

12. Используйте git log --oneline --graph --all, чтобы увидеть, что ваша ветка указывает на новый коммит, и что теперь у двух веток разные коммиты.

```
* 2b47d27 (HEAD -> main) file2.txt
| * b0e7d3b (mybranch) txt file
|/
* 989e637 (origin/main, origin/HEAD) aaaaaaaaaaa
* 8221493 aaaaa
* e95a8b6 Сообщение коммита: что поменялось
* d44bf5b Update and rename lab2.c to linked_lists.c
* 4cde8f6 Add files via upload
* 51cd71f Add files via upload
```

13. Переключитесь на вашу ветку mybranch.

14. Наш file2.txt пропал? Yos

```
* b0e7d3b (HEAD -> mybranch) txt file
* 989e637 (origin/main, origin/HEAD) aaaaaaaaaaa
* 8221493 aaaaa
* e95a8b6 Сообщение коммита: что поменялось
* d44bf5b Update and rename lab2.c to linked_lists.c
* 4cde8f6 Add files via upload
* 51cd71f Add files via upload
```

15. Используйте git diff mybranch master, чтобы увидеть разницу между двумя ветками.

```
diff --git a/file2.txt b/file2.txt
new file mode 100644
index 0000000..4a03c87
--- /dev/null
+++ b/file2.txt
@@ -0,0 +1 @@
+Second file
\ No newline at end of file
diff --git a/git_moves/file1.txt b/git_moves/file1.txt
deleted file mode 100644
index e9aaa42..0000000
--- a/git_moves/file1.txt
+++ /dev/null
@@ -1 +0,0 @@
-Ann Korotkova
```

16. Добавить текстовый документ со скриншотами в ветку mybranch. Закоммитить и запушить на удаленный репо ветку mybranch (git push -u origin mybranch)

17. Убедиться, что в github.com две ветки master и mybranch. Не забыть запушить изменения master ветки в master. Допустим

Оценка 4

1. Переключитесь на ветку mybranch. В ней будет файл sort.c из предыдущих шагов с функцией сортировки.

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/пpora/C2-433/git_moves$ git log --oneline --grap
* 33450b7 (HEAD -> mybranch, origin/mybranch) допустим
* b0e7d3b txt file
* 989e637 (origin/main, origin/HEAD) aaaaaaaaaaa
* 8221493 aaaaa
* e95a8b6 Сообщение коммита: что поменялось
* d44bf5b Update and rename lab2.c to linked_lists.c
* 4cde8f6 Add files via upload
* 51cd71f Add files via upload
```

2. Перезапишите содержимое в sort.c добавив функцию main(), в которой будет объявлен массив из нескольких чисел (пример int a[] = {4, 2, 0};) и вызвана функция сортировки для этого массива.
3. Что вам говорит git diff? Показывает че добавилось, че убавилось
4. Что вам говорит git diff --staged? Пустой? Пустой, потому что в индексе нет изменений, которые отличаются от текущего состояния коммита
5. Добавьте в staged файл sort.c
6. Что вам говорит git diff? Ничего
7. Что вам говорит git diff --staged? Ничего
8. Удалите любое из чисел в массиве в sort.c:
9. Что вам говорит git diff? Изменение
10. Что вам говорит git diff --staged? Ничего
11. Объясните, что происходит. **Не знаю, видимо git diff показывает различия между рабочей директорией и индексом. А git diff --staged показывает различия между индексом и последним коммитом**
12. Запустите git status и обратите внимание, что sort.c присутствует дважды в выводе. Ну и прикол, реально

```
On branch mybranch
Your branch is up to date with 'origin/mybranch'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   sort.c

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   sort.c
```

13. Запустите git restore --staged sort.c, чтобы отменить индексацию изменения
14. Что вам теперь говорит git status?

```
On branch mybranch
Your branch is up to date with 'origin/mybranch'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   sort.c

no changes added to commit (use "git add" and/or "git commit -a")
```

15. Индексируйте изменение (add) и сделайте коммит
16. Как выглядит журнал?

```
commit b262ce9b3d09d8baa456dd9480aba1d6a7e3c17c (HEAD -> mybranch)
Author: annkits <annpodolskaya0965@gmail.com>
Date:   Sun Mar 9 11:16:02 2025 +0700

    15 commit
```

17. Добавьте в sort.c в main() printf(“hello git\n”);.
18. Каково содержимое sort.c?

```
//aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

void SelectionSort(struct Student arr[]){
    for (int i = 0; i < size - 1; i++){
        int j_max = i;
        for (int j = i + 1; j < size; j++){
            if (arr[j].total > arr[j_max].total){
                j_max = j;
            }
        }
        struct Student temp = arr[i];
        arr[i] = arr[j_max];
        arr[j_max] = temp;
        memset(&temp, 0, sizeof(struct Student));
    }
}

int main(){
    int a[] = {4, 2};
    int b = SelectionSort(a);
    printf(“hello git\n”);      You, 1 second ago • Uncommitted changes
}
```

19. Что нам говорит git status?

```
On branch mybranch
Your branch is ahead of 'origin/mybranch' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   sort.c

no changes added to commit (use "git add" and/or "git commit -a")
```

20. Запустите git restore sort.c
21. Каково содержимое sort.c?

```
int main(){
    int a[] = {4, 2};
    int b = SelectionSort(a);      annkits, 2 minutes ago • 15 commit
}
```

22. Что нам говорит git status?

```
On branch mybranch
Your branch is ahead of 'origin/mybranch' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

23. Запушить на удаленный репо ветку.

Ветки и ff-merge

1. Создать файл greeting.txt, проиндексировать его и закоммитить с сообщением “Add file greeting.txt”.

```
[mybranch 22733f1] Add file greeting.txt
 1 file changed, 1 insertion(+)
 create mode 100644 git_moves/greeting.txt
```

2. Добавить в этот файл слово hello, индексируем и коммитим с текстом "Add content to greeting.txt"
3. Создайте ветку с именем feature/uppercase (да, feature/uppercase — это совершенно допустимое имя ветки и общепринятое соглашение).
4. Переключитесь на эту ветку

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/npora/C2-433/git_moves$ git switch feature/upper
Switched to branch 'feature/uppercase'
```

5. Каков вывод git status?

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/прога/С2-433/git_moves$ git status
On branch feature/uppercase
nothing to commit, working tree clean
```

6. Отредактируйте greeting.txt, чтобы он содержал приветствие в верхнем регистре (HELLO)
7. Добавьте файл greeting.txt и закоммитьте

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/прога/С2-433/git_moves$ git commit -m "HEELOW"
[feature/uppercase 94950a1] HEELOW
 1 file changed, 1 insertion(+), 1 deletion(-)
```

8. Каков вывод git branch?

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/прога/С2-433/git_moves$ git branch
* feature/uppercase
  main
  mybranch
```

9. Каков вывод git log --oneline --graph –all

```
fatal: ambiguous argument '–all': unknown revision or path not in the working tree.
Use '--' to separate paths from revisions, like this:
'git <command> [<revision>...] -- [<file>...]'
```

10. Переключитесь на главную ветку

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/прога/С2-433/git_moves$ git switch main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
```

11. Используйте cat, чтобы увидеть содержимое файла greetings.txt

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/прога/С2-433/git_moves$ cat greeting.txt
cat: greeting.txt: No such file or directory
```

12. Сравните ветки

```
diff --git a/file2.txt b/file2.txt
deleted file mode 100644
index 4a03c87..0000000
--- a/file2.txt
+++ /dev/null
@@ -1 +0,0 @@
-Second file
\ No newline at end of file
diff --git a/git_moves/chronology.docx b/git_moves/chronology.docx
deleted file mode 100644
index e69de29..0000000
diff --git a/git_moves/chronology.txt b/git_moves/chronology.txt
new file mode 100644
index 0000000..d828477
--- /dev/null
+++ b/git_moves/chronology.txt
@@ -0,0 +1 @@
+допустим, здесь должны быть скриншоты...
\ No newline at end of file
diff --git a/git_moves/file1.txt b/git_moves/file1.txt
new file mode 100644
index 0000000..e9aaa42
--- /dev/null
+++ b/git_moves/file1.txt
@@ -0,0 +1 @@
```

13. Объедините ветки
14. Используйте cat, чтобы увидеть содержимое файла greetings.txt

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/пpора/C2-433/git_moves$ cat greeting.txt
HEELOW
hallou
```

15. Удалите ветку с заглавными буквами (feature/uppercase)

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/пpора/C2-433/git_moves$ git branch -D feature/u
Deleted branch feature/uppercase (was 94950a1).
```

16. Смержить ветку mybranch в master (git merge)

```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/пpора/C2-433/git_moves$ git merge --ff mybranch
Already up to date.
```

17. Что выводит git log --oneline --graph –all?

```
*   0f7e7d7 (HEAD -> main) conflict
|\
| * 94950a1 HEELOW
| * affd319 (mybranch) Add content to greeting.txt
| * 934aa64 Add file greeting.txt
*   0f7e7d7 (HEAD -> main) conflict
|\
| * 94950a1 HEELOW
| * affd319 (mybranch) Add content to greeting.txt
| * 934aa64 Add file greeting.txt
| * 22733f1 Add file greeting.txt
| * b262ce9 (origin/mybranch) 15 commit
*   0f7e7d7 (HEAD -> main) conflict
|\
| * 94950a1 HEELOW
| * affd319 (mybranch) Add content to greeting.txt
```

18. Запушить изменения ветки master на удаленный репо.

19. Запушить документ с результатами вашей работы

Оценка 5

1. Создать ветку branch1, переключиться на нее
```
git branch branch1
```
```
git switch branch1
```
2. Выполнить команду

   $ echo "This is a relevant fact" > file.txt
```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/npora/C2-433/git_moves$ echo "This is a releva
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/npora/C2-433/git_moves$ ls
chronology.txt  file.txt  file1.txt  greeting.txt  sort.c
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/npora/C2-433/git_moves$ cat file.txt
This is a relevant fact
```
3. Закоммитить это изменение
```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/npora/C2-433/git_moves$ git commit -m "fact"
[branch1 56f9d7e] fact
 1 file changed, 1 insertion(+)
 create mode 100644 git_moves/file.txt
```
4. Переключиться на главную ветку и выполнить команду

   echo "This is an indispensable truth!" > file.txt
5. Закоммитить изменения в master
```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/npora/C2-433/git_moves$ git commit -m "truth"
[main 15cfa49] truth
 1 file changed, 1 insertion(+)
 create mode 100644 git_moves/file.txt
```
6. Каков вывод git log --oneline --graph –all
```
* 15cfa49 (HEAD -> main) truth
| * 56f9d7e (branch1) fact
|/
```
7. Использовать команду git merge чтобы смержить ветку branch1 в master (получим конфликт это норм)
```
korotkova@LAPTOP-EME5GNUM:/mnt/c/Users/annpo/OneDrive/Desktop/учёба/npora/C2-433/git_moves$ git merge branch1
Auto-merging git_moves/file.txt
CONFLICT (add/add): Merge conflict in git_moves/file.txt
Automatic merge failed; fix conflicts and then commit the result.
```

8. Что показывает git status?

```
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both added:      file.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

9. Посмотреть содержимое файла file.txt и в любом любимом текстовом редакторе исправьте конфликт.
10. Что показывает git log --oneline –graph?

```
*   fecf16b (HEAD -> main) 1try
|\
| * 56f9d7e (branch1) fact
* | 15cfa49 truth
|/
```

11. Запушить изменения

Merge конфликты для сортировки MergeSort на python.

1. Находясь в ветке master создадим файл mergesort.py с содержимым из base.py
2. Проиндексируем файл и закоммитим

```
[main 1b5a811] mergesort
 1 file changed, 5 insertions(+)
 create mode 100644 git_moves/mergesort.py
```

3. Создадим новую ветку Mergesort-Impl и переключимся на нее.
4. Содержимое файла mergesort.py заменим на код из righty.py
5. Коммитим изменения.

```
[Mergesort-Impl cf0d810] righty
 1 file changed, 89 insertions(+), 5 deletions(-)
 rewrite git_moves/mergesort.py (98%)
```

6. Переключаемся на master и меняем все содержимое mergesort.py на lefty.py

7. Коммитим изменения.

```
[main b39462e] lefty
 1 file changed, 89 insertions(+), 5 deletions(-)
 rewrite git_moves/mergesort.py (98%)
```

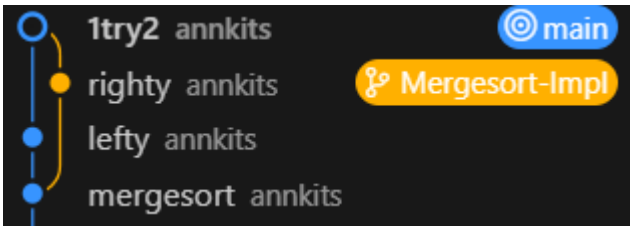
8. Что показывает git log --oneline –graph?

```
* b39462e (HEAD -> main) lefty
* 1b5a811 mergesort
```

9. Что показывает git branch?

```
  Mergesort-Impl
  branch1
* main
  mybranch
```

10. Необходимо смержить Mergesort-Impl в master.
11. После исправления всех merge конфликтов запушить в master изменения.



12. Запушить документ с результатами вашей работы