

Лабораторна робота №6. Масиви

Індивідуальне завдання на оцінку “добре” номер 2.

1. Вимоги

1) Розробник

- Князькін Владислав Ігорович
 - студент групи КІТ-320
- 23.11.20

2) Загальне завдання

Заповнити масив цілих чисел заданного розміру чисел Фіббоначчі.

2. Хід роботи

1. Зробимо у раніше створеному репозиторії нову під-директорію *lab06*.

```
valadon@valadon-VirtualBox:~ PROGRAMMING-KNIAZKIN/naxxramass/PROGRAMMING-KNIAZKIN $ mkdir lab06
```

2. Створимо папку для нашого завдання *task02*, у якому створимо папку *src* та відкриємо і запустимо у ньому *nano*.

```
valadon@valadon-VirtualBox:~ PROGRAMMING-KNIAZKIN/naxxramass/PROGRAMMING-KNIAZKIN/naxxramass/PROGRAMMING-KNIAZKIN/lab06/ $ mkdir task02
valadon@valadon-VirtualBox:~ PROGRAMMING-KNIAZKIN/naxxramass/PROGRAMMING-KNIAZKIN/naxxramass/PROGRAMMING-KNIAZKIN/lab06/task02/src $ nano
```

3. Записуємо наш код програми, дотримуючись всіх відступів та закриття дужок.

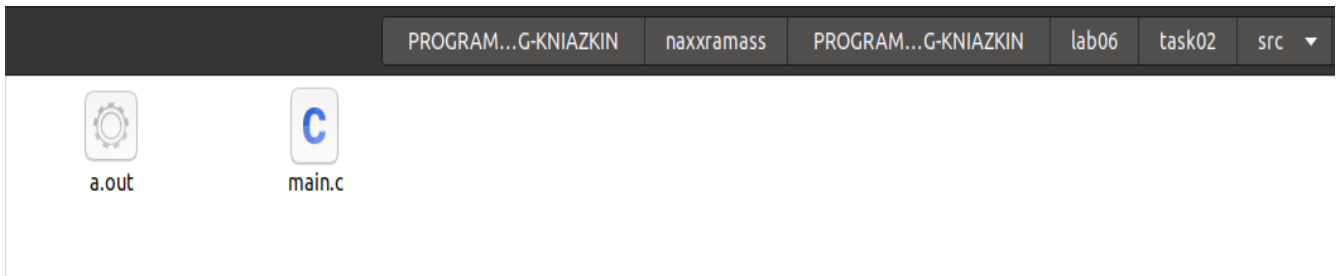
Запишемо нашу програму:

```
int main () { // * створюємо функцію
int a [20]; // * створюємо масив з 20 елементів
int i; // * створюємо змінну для нумерації елементів
a [0] = 0; // * оголошуємо початкове значення нульового елемента масиву
a [1] = 1; // * оголошуємо початкове значення першого елемента масиву
for (i = 0; i <20; i ++) { // * створюємо цикл для заповнення масиву цілими числами Фіббоначі
a [i] = a [i - 1] + a [i - 2]; // * записуємо формулу для обчислення чисел Фіббоначі
} // * кінець циклу
return 0; // * Закінчуємо поточну функцію main () і повертаємо результат програми
} // * кінець функції
```

Збережемо наш код та підпишемо його *main.c*.

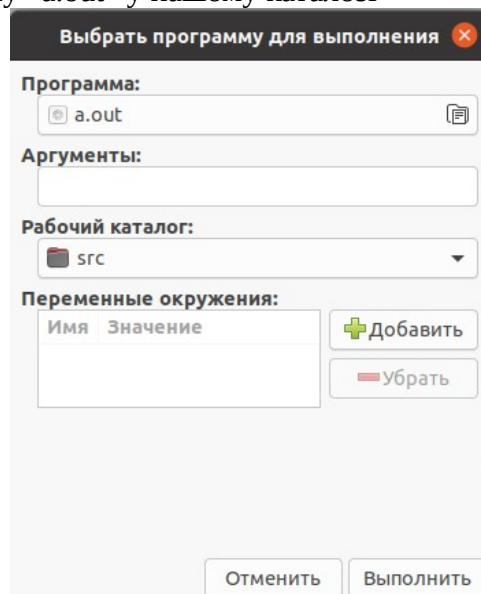
4. Повертаємось до терміналу. Завдяки команді (у каталозі *src*):

```
gcc -g main.c /* Зробимо виконувальні файли для наших main.c
Вони автоматично назвуться a.out
```



5. Запускаємо відлагодник (nemiver), щоб перевірити, як працює наша програма. Щоб запустити програму, треба запустити її через виконувальний файл:

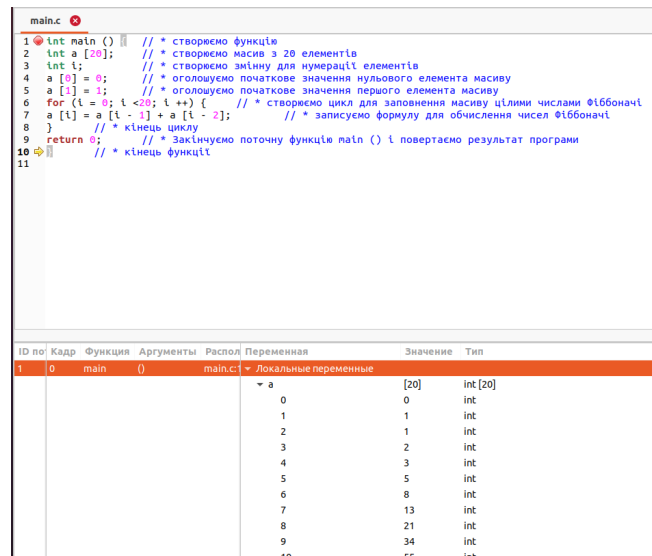
1. Відкриваємо “Файл”
2. Знаходимо функцію “Завантажити виконувальний файл...”
3. Вибираємо програму “a.out” у нашому каталозі



4. Натискаємо “Виконати”.

Тепер можна побачити програму, яку ми написали раніше:

Завдяки функції “Step over”, переводимо нашу “стрілку” до *останнього рядка*, щоб перевірити результат команди.



6. Бачимо, що результат вірний.

Видалимо a.out. Збережемо зміни у нашій директорії на *github* через команди *git*:

```
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass $ git add .
```

```
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass $ git status
```

```
valadon@valadon-VirtualBox: ~/PROGRAMMING-KNIAZKIN/na...
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass$ git add .
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass$ git status
На ветке main
Ваша ветка обновлена в соответствии с «origin/main».

Изменения, которые будут включены в коммит:
  (use "git restore --staged <file>..." to unstage)
    переименовано: PROGRAMMING-KNIAZKIN/lab05/lab05_4/for/a.out -> PROGRAMMI
NG-KNIAZKIN/lab04/task01/src/a.out
    изменено:      PROGRAMMING-KNIAZKIN/lab04/task01/src/main.c
    удалено:      PROGRAMMING-KNIAZKIN/lab05/lab05_4/for/main.c
    удалено:      PROGRAMMING-KNIAZKIN/lab05/lab05_4/while_do/main.c
    новый файл:   PROGRAMMING-KNIAZKIN/lab05/task01/README.md
    новый файл:   PROGRAMMING-KNIAZKIN/lab05/task02/README.md
    новый файл:   PROGRAMMING-KNIAZKIN/lab05/task03/README.md
    новый файл:   PROGRAMMING-KNIAZKIN/lab05/task05/README.md
    переименовано: PROGRAMMING-KNIAZKIN/lab05/lab05_4/while_do/a.out -> PROG
RAMMING-KNIAZKIN/lab05/task05/src/a.out
    новый файл:   PROGRAMMING-KNIAZKIN/lab05/task05/src/main.c
    удалено:     PROGRAMMING-KNIAZKIN/lab06/lab06_1/main.c
    удалено:     PROGRAMMING-KNIAZKIN/lab06/lab06_2/a.out
    удалено:     PROGRAMMING-KNIAZKIN/lab06/lab06_2/main.c
    новый файл:   PROGRAMMING-KNIAZKIN/lab06/task01/README.md
    новый файл:   PROGRAMMING-KNIAZKIN/lab06/task01/src/main.c
```

```
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass $ git commit -m
"Create lab06_task02"
```

```
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass $ git push
```

```
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass$ git push
Username for 'https://github.com': Korv3L
Password for 'https://Korv3L@github.com':
Перечисление объектов: 50, готово.
Подсчет объектов: 100% (50/50), готово.
Сжатие объектов: 100% (34/34), готово.
Запись объектов: 100% (38/38), 9.18 KiB | 626.00 KiB/s, готово.
Всего 38 (изменения 3), повторно использовано 0 (изменения 0)
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
To https://github.com/Korv3L/naxxramass
  08b46ef..a45448d  main -> main
valadon@valadon-VirtualBox:~/PROGRAMMING-KNIAZKIN/naxxramass$
```

3. Способи використання

Числа Фібоначчі знайшли собі використання у багатьох сферах людської культури, наприклад у мистецтві(Сітка Фібоначчі або

“Золотий перетин”), **природі**(листорозміщення, розташування насіння та осередків ананасу, відношення частин тіла людини), **бізнесі**(послідовність Фібоначчі є інструментом технічного аналізу, що використовуються професійними трейдерами в поєднанні з іншими інструментами для розрахунку прогнозу потенційного кінця корекції, приймаючи відсоток від попереднього руху) та навіть **кодуванні**(Фібоначчієва система числення - змішана система числення для цілих чисел на основі чисел Фібоначчі $F2 = 1, F3 = 2, F4 = 3, F5 = 5, F6 = 8$ і т. д.).

4. Висновок

Завдяки цій лабораторній роботі, я навчився робити звичайні операції з масивами на язику C, у системі Linux, додавати до них коментарі, перевіряти програму на дієздатність у відлагоднику (nemiver), та відправляти зміни у свій репозиторій на github.