Лабораторна робота №7. Функції

Автор: Барчан Іван
Група: КН-922Б

Завдання:

1.Переробити програми, що були розроблені під час виконання лабораторних робіт з тем "Масиви" та "Цикли" таким чином, щоб використовувалися функції для обчислення результату.

2.Функції повинні задовольняти основну їх причетність - уникати дублювання коду.

Тому, для демонстрації роботи, ваша програма (функція main()) повинна мати можливість викликати розроблену функцію з різними вхідними даними.

3.Слід звернути увагу: параметри одного з викликів функції повинні бути згенеровані за допомогою генератора псевдовипадкових чисел random().

4.Слід звернути увагу (#2): продемонструвати встановлення вхідних даних через аргументи додатка (параметри командної строки).

Обробити випадок, коли дані не передались - у цьому випадку вони матимуть значення за умовчуванням, обраними розробником.

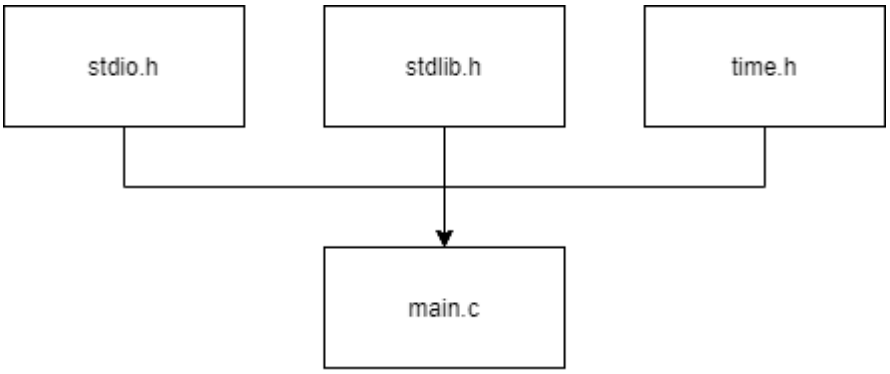
Опис програми

Функціональне призначення

Ця програма виконує 1 з 2 функцій яку обирає користувач.

- З’ясовує чи є правильною цифра
- Множить матрицю саму на себе

Опис логічної структури



(Рис. 1) Графічна структура програми

Файл "main.c"

Головний файл

Це файл, який містить точку входу, виклики функцій primenumber , matrix та значення для аргументів цих функцій.

main()

Головна функція

Послідовність дій

Спочатку введіть номер операції яку ви хочете виконати, для цього :

Впишемо значення аргументу x.

x - аргумент типу int необхідний для того щоб виконати одну з наступних дій :

- Якщо користувач ввів 1 виконується знаходження простої цифри
- Якщо користувач ввів 2 виконується знажодження результату від множення матриці саму на себе
- Якщо користувач ввів будь яку іншу цифру програма зупиняється.

Якщо виконується знаходження простої цифри то користувач повинен ввести цифру яка його цікавить. Для цього потрібно ввести аргумент n.

n - аргумент типу int, який є цифрою яку користувач хоче перевірити на простоту.

- Якщо цифра яку ввів користувач не є нулем то може виконуватись функція primenumber.
- Якщо цифра введена цифра нуль то генерується випадкова цифра та може виконуватись функція primenumber.

Якщо виконується знаходження результату від множення матриці саму на себе, користувач повинен ввести спочатку кількість рядків, а потім кількість стовпців.

Для цього потрібно ввести аргументи b та c, а потім, якщо виконується умова, ввести саму матрицю, множення якої цікавить користувача.

b - аргумент типу int, який означає кількість рядків.

c - аргумент типу int, який означає кількість стовпців.

i та j-кількість стовпців і рядків матариці, які порівнюються між заданими b та c, та якщо виконується умова вони збільшуються.

a[10][10]-квадратна матриця, що містить межу 10 рядків і стовпців, але користувач може задати будь-яку квадратну матрицю в цьому діапазоні.

- Якщо кількість рядків та стовпців співпадає програма виконується, якщо ні, то не виконується.
- Якщо кількість рядків та стовпців дорівнюють нулям, то матриця задається розробником, це квадратна матриця в якої всі цифри двійки.
- Якщо кількість рядків та стовпців не дорівнюють нулям, то матрицю заповнює користувач.

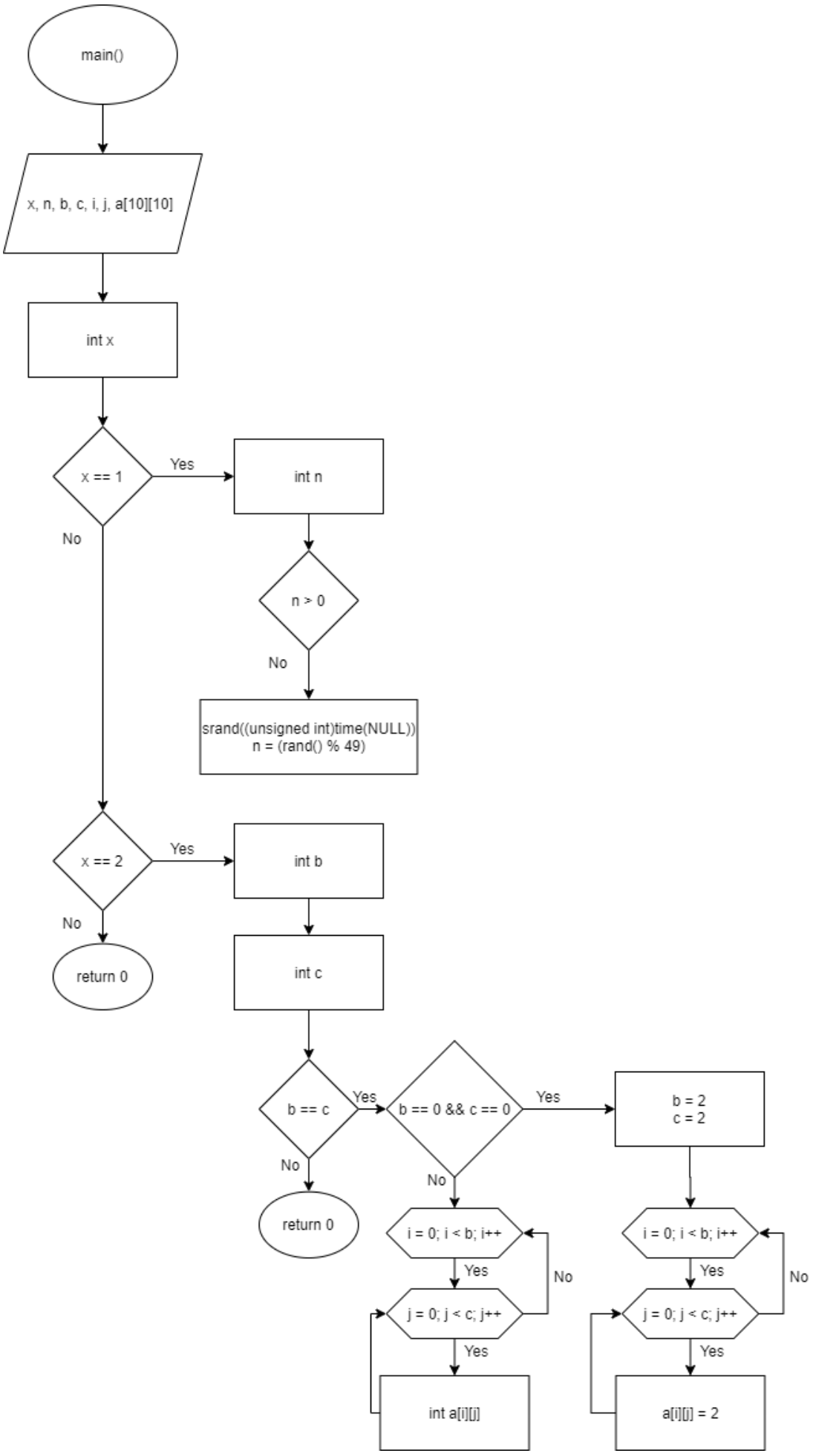
Після того як користувач все ввів може виконуватись функція matrix.

```
int main()
{
   int x, n, b, c, i, j, a[10][10] ;


   scanf("%d",&x);
      if (x == 1)
      {
         scanf("%d",&n);
         if (n > 0)
         {


         }
         else
         {
            srand((unsigned int)time(NULL));
            n = (rand() % 49);
         }
      }
      else if (x == 2)
      {
         scanf("%d", &b);
            scanf("%d", &c);
            if (b == c)
            {
               if (b == 0 && c == 0)
```

```
            {
               b = 2 ;
               c = 2 ;
                for (i = 0; i < b; i++)
                  {
                        for (j = 0; j < c; j++)
                        {
                              a[i][j] = 2 ;
                        }
                  }
            }
            else
            {
                  for (i = 0; i < b; i++)
                  {
                        for (j = 0; j < c; j++)
                        {
                              scanf("%d", &a[i][j]);
                        }
                  }
            }
         }
         else
         {
               return 0;
      }
   }
   else
   {
      return 0;
   }
}
```



(Рис. 2) Схема алгоритму функції main

int primenumber(int n)

Ця функція з’ясовує чи є цифра простою

Послідовність дій

Додаємо змінну result.

Змінна result означає :

- Якщо result = 2 - цифра ні проста, ні не проста.
- Якщо result = 1 - цифра проста.
- Якщо result = 0 - цифра не проста.

Якщо n = 1, то result = 2.

Якщо n = 2, то result = 1.

Якщо n дорівнює будь якій цифрі крім 1 та 2, то запускається цикл який ділить дану цифру n на всі цифри починаючи з 2 і до n-1 :

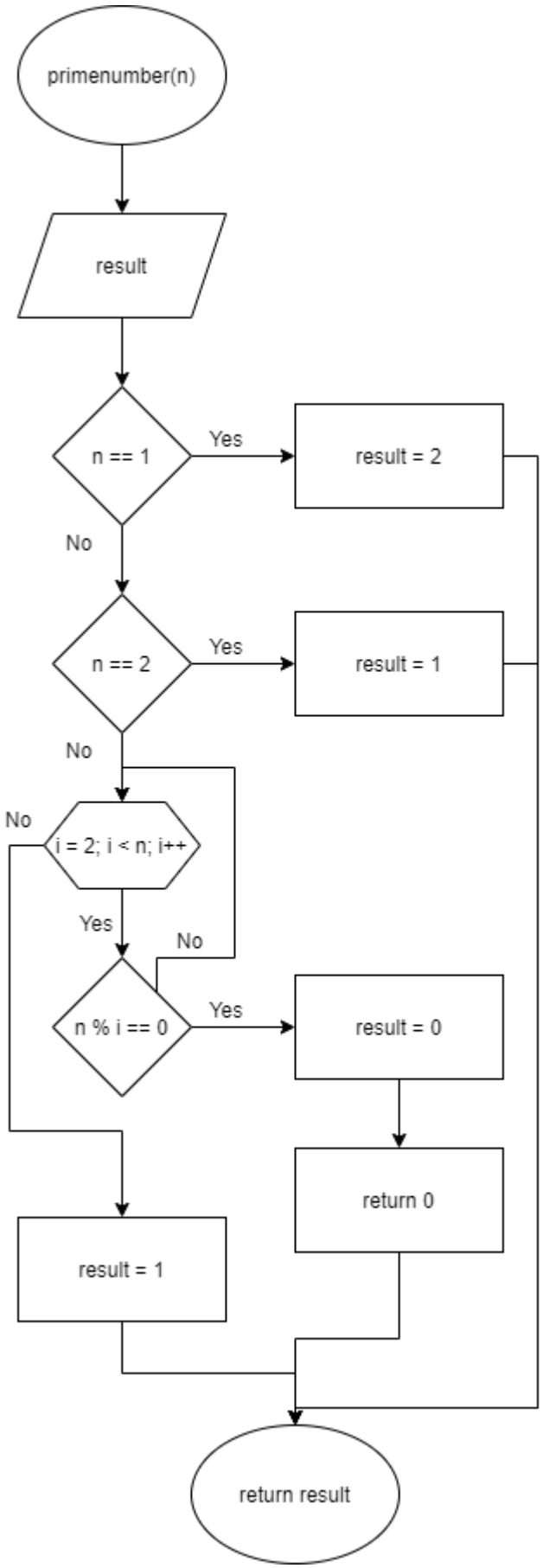
- Якщо хоч одна цифра ділиться націло, записуємо що result = 0 (цифра не проста).
- Якщо жодна цифра не ділиться націло записуємо що result = 1 (цифра проста).

Повертаємо змінну result.

```
int primenumber(int n)
   {
      int result;

      if (n == 1)
      {
         result = 2;
      }
      else if (n == 2)
      {
         result = 1;
      }
      else
      {

         for (int i = 2; i < n; i++)
         {
            if (n % i == 0)
            {
               result = 0;
               return 0;
            }
         }
         result = 1;
      }
   return result;
   }
```



(Рис. 3) Схема алгоритму функції primenumber

int matrix(int b, int c, int i, int j, int a[10][10])

Ця функція множить матрицю сама на себе

Послідовність дій

Додаємо матрицю MAT[10][10] та змінну f.

MAT[10][10]-квадратна матриця, що містить розрахунок матриці a*a.

f - це змінна типу int, що допомагає нам помножити саме рядок матриці на стовпчик

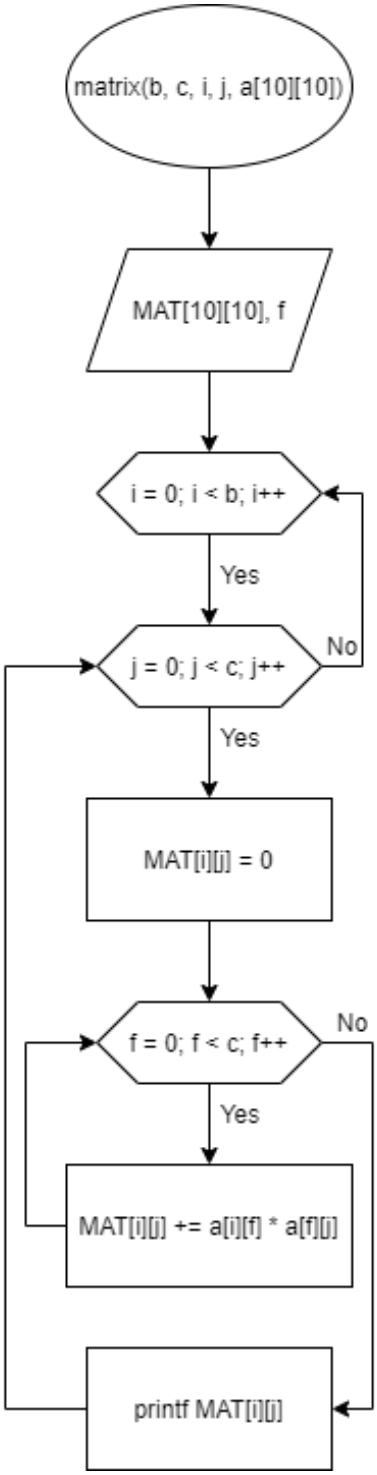
Перебираємо значення записані записані користувачем та множимо рядки матриці на стовпчики

Результат записуємо в MAT[i][j], та виводимо його.

```
int matrix(int b, int c, int i, int j, int a[10][10])
  {
      int MAT[10][10], f;

            for (i = 0; i < b; i++)
            {
                  for (j = 0; j < c; j++)
                  {
                        MAT[i][j] = 0;
                        for (f = 0; f < c; f++)
                        {
                              MAT[i][j] += a[i][f] * a[f][j];

                        }
                  printf ("%d\n", MAT[i][j]);
                  }
            }
  }
```



(Рис. 4) Схема алгоритму функції matrix

Структура проекту лабораторної роботи:

```
├── lab07
├── Makefile
├── README.md
└── src
    └── main.c
```

Варіанти використання

Користуватися цією програмою не складно. Для того щоб з’ясувати проста цифра чи ні вам потрібно :

- Ввести номер операції (Щоб дізнатись чи проста цифра введіть 1)
- Ввести цифру яка вас цікавить.

```
one4k@one4k-VirtualBox:~/programing-barchan/lab07/src$ ./a.out
1
41
```

(Рис. 5)Як правильно користуватися програмою!

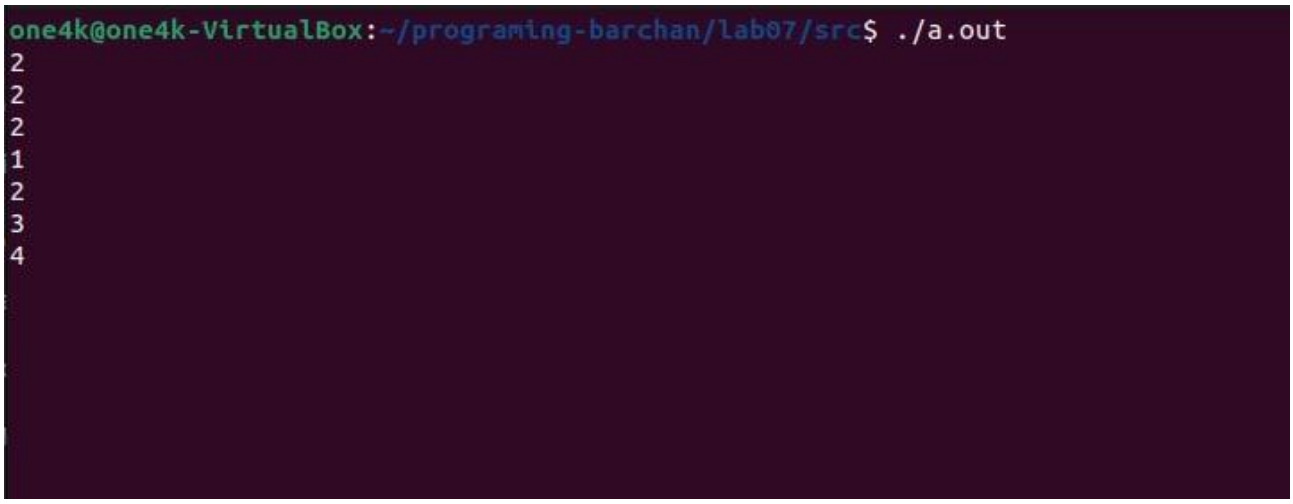
Щоб побачити результати роботи програми, вам потрібно завантажити її в LLDB, ввести що сказано вище та на рядку 66 написати p primenumber(n) (Вивести функцію яка з’ясовує чи проста цифра чи ні).

```
(lldb) n
Process 3192 stopped
* thread #1, name = 'a.out', stop reason = step over
    frame #0: 0x0000555555555360 a.out`main at main.c:66:1
   63          {
   64              return 0;
   65          }
-> 66      }
   67
   68
   69      int primenumber(int n) //Функція знаходження простої цифри
(lldb) p primenumber(n)
(int) $0 = 1
```

(Рис. 6) Як дізнатися чи проста цифра чи ні !

Для того щоб знайти результат множення матриці саму на себе вам потрібно :

- Ввести номер операції (Щоб знайти результат множення матриці введіть 2)
- Ввести кількість рядків та стовпців (Кількість рядків та стовпців повинна співпадати, інакше програма не буде виконуватись)
- Ввести саму матрицю



(Рис. 7) Як правильно користуватися програмою!

Щоб побачити результати роботи програми, вам потрібно завантажити її в LLDB, ввести що сказано вище та на рядку 61 написати p matrix(b, c, i, j, a) (Вивести функцію яка знаходить результат множення матриці саму на себе).

(Рис. 8) Як дізнатися результат множення матриці саму на себе !

Висновки: У цій роботі було перетворено лабораторні проекти №5 та №6 для використання функцій. Було набуто навичок роботи з функціями, їх декларація, реалізація та виклик. Під час тестування програми були отримали результати роботи функції matrix – це множення матриці саму на себе, і результати роботи функції primenumber - це перевірка цифри на те проста вона чи ні.