Лабораторна робота №5. Функції

Мета і задачі:

Навчитися створювати та відлагоджувати програми, в яких використовуються функції на мові програмування С.

Теоретичні відомості і методичні вказівки

Функція - це група операторів, що виконується, коли він викликається в якийсь момент програми. Формат функції наступний:

<ім'я типу> <назва функції>(параметр1, параметр2, ...)
{тіло функції}

Приклад програми з функцією, яка здійснює додавання двох чисел:

```
#include <stdio.h>


int addition (int a, int b)
{
  int r;
  r=a+b;
  return (r);
}
int main()
{
  int z;
  z = addition(5,3);
  printf("Функція повертає значення %i", z);
  return 0;
}
```

Функція *main* починається з оголошення змінної *z* типу *int*. Відразу після цього здійснюється виклик функції **addition**. Схожість між структурою виклику функції та оголошенням самої функції зображена на рис. 1:

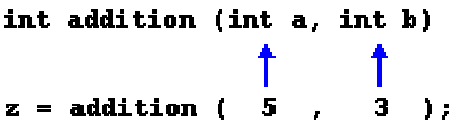

Рис 1.

У функції main здійснюється виклик функції *addition*, їй передаються через параметри два значення: 5 і 3, які відповідають параметрам *int a і int b* в оголошенні функції *addition*.

В точці, в якій функція *addition* викликається з *main* управління передається функції *addition*. Значення обох аргументів (5 і 3) будуть скопійовані в локальні змінні *int a* і *int b* в межах функції *addition*.

Функція *addition* оголошує локальну змінні *int r,* виконує вираз *r=a+b*, в результаті чого у r буде присвоєно значення 8, так як значення параметрів *a і b*, були 5 і 3 відповідно.

Наступний рядок коду:

```
return (r);
```

завершує функцію *addition*, і повертає значення назад у функцію main. Далі програма виконується з того ж місця, на якому вона була перервана викликом функції *addition*. Значення що повертається з функції є значенням функції, що викликається (рис 2.). Щоб пояснити це інакше, можна собі уявити, що виклик функції addition (5,3) буквально замінюється значенням я, яке вона повертає (8).

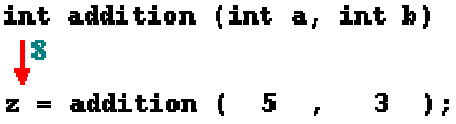

Рис. 2

Наступний рядок коду виводить на екран значення змінної z:

```
printf("Функція повертає значення %i", z);
```

Порядок виконання і звітування

1. Написати програму, яка демонструє роботу функції заданої відповідно до варіанту використавши середовище програмування Dev-C++ 4.0.
2. Відкомпілювати та відлагодити програму.
3. Розробити набір тестів і перевірити роботу програми на них.
4. Відповісти на контрольні запитання.
5. Зробити висновки.
6. Звіт по лабораторній роботі має складатися з титульної сторінки, лістингів програм, висновків по роботі.

Варіанти завдань

Варіант 1.

Як відомо, властивість послідовності чисел Фібоначі: 0, 1, 1, 2, 3, 5, 8, 13, 21, ... полягає у тому, що кожне наступне число є сумою двох попередніх.

Написати нерекурсивну функцію **fibonacci (n)**, що обчислює n-не число Фібоначі.

Варіант 2.

Найбільший загальний дільник (НОД) двох цілих чисел є самим більшим цілим числом, на яке ділиться кожне із двох чисел.

Написати функцію **gcd,** яка повертає найбільший загальний дільник двох цілих чисел.

Варіант 3.

Найбільший загальний дільник цілих чисел *x* і *y* є найбільшим цілим числом, на яке діляться і *x* і *y*.

Написати рекурсивну функцію **gcd,** що повертає найбільший загальний дільник *x* і *y* та програму, яка використовує цю функцію. Рекурсивне визначення функції **gcd** виглядає в такий спосіб: якщо *y* дорівнює 0**,** то *gcd(x, y)* є *x*, інакше *gcd(x, y)* є *gcd(y, x % y)*, де % — операція ділення по модулю.

Варіант 4

Ціле число називають простим, якщо воно ділиться тільки на 1 і на саме себе. Наприклад, 2, 3, 5 і 7 є простими, а 4, 6, 8 і 9 - немає.

Написати функцію, яка визначає, чи є число простим. Використайте цю функцію в програмі, що знаходить і друкує всі прості числа в діапазоні від 1 до 10000.

Варіант 5.

Ціле число називають досконалим числом, якщо сума його дільників, включаючи 1 (але не саме число), дорівнює цьому числу. Наприклад, 6 є досконалим числом, оскільки 6 = 1+ 2 + 3.

Написати функцію **perfect,** яка визначає, чи є її параметр досконалим числом. Використайте цю функцію в програмі, що знаходить і друкує всі досконалі числа в діапазоні від 1 до 1000. Надрукуйте всі дільники для кожного досконалого числа, щоб переконатися, що число дійсно є досконалим.

Варіант 6.

Написати функцію, яка одержує час у якості трьох цілих аргументів (години, хвилини й секунди) і повертає число секунд із моменту, коли годинник «пробив 12».

Варіант 7.

Написати функцію, яка одержує ціле значення і повертає число із оберненим порядком цифр. Наприклад, для числа 7631 функція повинна повернути значення 1367.

Варіант 8.

Написати функцію **gualityPoints**, яка вводить середній бал студента й повертає 4, якщо середній бал студента становить 90-100, 3, якщо 80-89, 2, якщо 70-79, 1, якщо 60-69, і 0, якщо середній бал нижче ніж 60.

Варіант 9.

Написати програму, яка моделює підкидання монети. Для кожного підкидання монети програма повинна друкувати слова «Heads» або «Tails» (орел або решка). Нехай програма підкине монету 100 разів і порахує число випадань для кожної сторони монети. Програма повинна викликати окрему функцію **flip,** що не одержує ніяких аргументів і повертає 0 для решки й 1 для орла.

Варіант 10.

Написати рекурсивну функцію **power (base, exponent)**, яка повертає значення $base^{exponent}$.

Наприклад, power(3, 4) = 3 * 3 * 3 * 3 . Нехай *exponent* є цілим, більшим або рівним 1.

Підказка: крок рекурсії міг би використати співвідношення $base^{exponent} = base * base^{exponent}$, а завершальною умовою буде випадок, коли значення *exponent* стане рівним 1.

Підсумок

Після виконання лабораторної роботи студент повинен вміти створювати програми, в яких використовуються функції на мові програмування С.

Контрольні питання

1. Що таке функція ?
2. Де і як визиваються функції ?
3. Що є аргументом функції ?

4. Написати загальний формат знаходження функції ?
5. Що може зробити визвана функція ?
6. Що таке прототип функції ?
7. Для чого призначені компілятори в функції ?
8. Де знаходяться прототипи функцій **rand** і **srand** , і що це таке ?

Контрольні вправи

1. Знайдіть помилку у фрагменті програми:

a.
```
int g (void)
 {
    printf (“Inside function g\n”);
    {
        printf(“Inside function h\n”);
    }
 }
```

b.
```
int sum (int x, int y)
 {
      int result;
      result = x + y;
```

c.
```
void f ( float a ) ;
 {
      float a ;
      printf ( “%f”, a ) ;   }
```

2. Напишіть функцію distance, яка обчислює відстань між двома точками з координатами (x1, y1) і (x2, y2). Всі числа і значення що повертаються повинні мати тип **float**.
3. Написати функцію, яка обчислює та повертає максимальне значення з двох переданих через параметри.
4. Написати функцію, яка обчислює та повертає мінімальне значення з двох переданих через параметри.
5. Підрахувати в одномірному масиві цілого типу розміром 100 елементів кількість нульових значень.

6. Написати функцію з двома параметрами логічного типу, який повертає значення у відповідності по наступній таблиці істинності (таблиця 1).

Таблиця 1

Параметри		Повернений результат
Перший	Другий	
False	False	False
False	True	False
True	False	True
True	True	False

Параметри і результати - цілого типу: false відповідає нулю і true - нульовим значенням.

Джерела інформації

1. http://void.net.ua/The_C_Programming_Language.html.
2. http://publications.gbdirect.co.uk/c_book/.
3. http://www.scribd.com/doc/16306895/Draft-ANSI-C-Rationale
4. http://www.cplusplus.com/doc/tutorial/