Лабораторна робота з дисципліни «Алгоритмізація та програмування» № 5

Тема: Використання функцій. Рекурсивні функції

Мета роботи: Вивчити засоби мови С для оголошення та визначення функцій, глобальних та локальних змінних, виклику функцій.

Теоретичні відомості

При розробці складних програм зручно програмну реалізацію окремих завдань (частин завдань) записувати як окремі функції.

Функція – це блок коду, який виконує певні дії. Функція має назву і може повертати певне значення як результат. Кожна функція має вирішувати лише одну конкретну задачу, ім'я функції має чітко описувати її призначення (зміст цієї задачі)*Функції корисні для групи команд в єдиному блоці, який можна використовувати багато разів*.

Функції можуть запускатися (викликатися) в коді програми будь-яку кількість разів. Значення, які передаються функції, називаються її аргументами. Їх типи мають відповідати типам параметрів у заголовку функції.

Визначення (опис) функції складається із заголовка функції, за яким записують тіло функції у фігурних дужках. Заголовок функції складається із типу результату функції, за яким записують ім'я функції, після імені функції в круглих дужках записують список параметрів функції. Загальний синтаксис функції наступний

```
<тип_результату> <ім'я_функції>([<список_параметрів_функції>])
 {
     <тіло_функції> //
 }
```

Список параметрів може містити елементи, відокремлені комою один від одного. Кожний елемент – це пара *тип – ім'я*: `<тип_параметру> <ім'я_параметру>/`

Параметри можуть бути відсутні – тоді після імені функції вказуються порожні круглі дужки. В блоці функції можна оголосити інші локальні змінні, як і у будь-якому блоці. Правила для імен параметрів – ті ж самі, що і для імен змінних. Тип результату функції може бути будь-яким, крім масиву та функції.

Локальні та глобальні змінні

Параметри функції – це її локальні змінні. Функції використовують пам'ять зі стека програми і після завершення роботи виділена для функції ділянка пам'яті звільняється.

Змінні, оголошені всередині функції або блоку називаються *локальними змінними*. Областю видимості локальних змінних є функція, в якій вони оголошені. Для решти функцій програми ці змінні та їхні значення є невідомими.

Глобальні змінні переважно визначаються у верхній частині програми перед усіма тілами функцій. Тобто глобальна змінна видима (доступна) всім функціям, оголошеним після неї, і всі вони можуть змінювати її значення. Оскільки, використання глобальних змінних часто призводить до помилок, тому слід намагатися уникати використання таких змінних.

Після оголошення кожну локальну змінну обов'язково потрібно ініціалізувати самостійно. Глобальні ж змінні ініціалізовуються автоматично.

Прототип функції складається із типу результату функції, за яким записують ім'я функції, після імені функції в круглих дужках записують список параметрів функції (може бути порожнім). Прототип функції завершується символом ; «крапка з комою»:

```
<тип_результату> <ім'я_функції>([<список_параметрів_функції>]);
```

Якщо в програмі є прототип деякої функції, то далі за текстом обов'язково слід навести повний опис (визначення) цієї функції. Списки параметрів і типи результатів в прототипі та визначенні функції мають відповідати один одному.

Виконання функції завершується після виконання команди *return* `<вираз>;`, або *return*`;`, або при досягненні фігурної дужки `}`, яка закриває блок функції.

Якщо *функція повертає результат*, то тип результату, вказаний перед іменем функції, має відповідати типу виразу, вказаного після ключового слова `return`. Тоді в тілі функції обов'язково має виконуватися команда `return <вираз>;`. Значення виразу, записаного в команді `return <вираз>;` – буде результатом функції.

Якщо *функція не повертає результату*, то замість типу результату в заголовку функції слід вказати ключове слово `void`. Команда `return;` в цьому випадку – необов'язкова.

Виклик функції завжди позначається ім'ям функції та круглими дужками, в яких стоять змінні (константи, вирази), значення яких передаються (підставляються) замість аргументів. У випадку, коли функція не має аргументів, круглі дужки все одно потрібно ставити.

В наступному прикладі визначено функцію, що має один параметр цілого типу та повертає квадрат заданого чила.

```
#include <iostream>
using namespace std;

// прототип функції
int square(int n);

// головна функція
int main()
{
	for (int i = 1; i <= 10; i++)
		cout << square(i) << endl; // виклик функції
	return 0;
	}
// реалізація функції
int square(int n)
{
```

```
	//тіло функції
	return n * n;//повертає ціле число
}
```

Прототип потрібний для того, щоб вказати компілятору, що `square()` – це функція. Якщо б не було прототипу, компілятор не знав би, що означає `square(i)` при виклику функції у наступному рядку:

```
–	cout << square(i) << endl;  // виклик функції
```

Для даного завдання можна було б використати функції, що не повертає результату наступним чином

```
#include <iostream>
using namespace std;

// визначення функції
void printSquare(int n)
{
	cout << n * n << endl;
}

// головна функція
int main()
{
	for (int i = 1; i <= 10; i++)
		printSquare(i); // виклик функції
	}
```

При виклику функції у першу чергу обчислюються вирази, які стоять на місці фактичних параметрів; потім у стеку виділяється пам’ять під формальні параметри функції відповідно до їхнього типу і кожному з них присвоюється значення відповідного фактичного параметра. При цьому перевіряється відповідність типів і, якщо необхідно, виконуються їхні перетворення. При невідповідності типів видається діагностичне повідомлення.

Хорошим стилем програмування вважається ніколи не використовувати глобальні змінні, якщо немає на те виняткової потреби.

Способи передавання параметрів до функцій

Область оперативної пам’яті, що використовується програмою, поділяється на сегмент коду, сегмент даних та сегмент стеку. У сегменті коду зберігаються команди програми, в сегменті даних – значення глобальних змінних, а в сегменті стеку – значення локальних змінних і параметрів підпрограм.

Є два способи, якими можна передати інформацію у функцію за допомогою параметрів: передавати значення аргументів; передавати адреси аргументів.

В мові C++ інформацію у функцію можна передавати за допомогою параметрів, які поділяються на наступні три види:

- параметри-значення *<тип_параметру> <ім'я_параметру>*
- параметри-вказівники *<тип_параметру> *<ім'я_параметру>*;
- параметри-посилання *<тип_параметру> &<ім'я_параметру>*.

За замовчуванням у мові C++, для передавання аргументів використовується виклик за значенням, при якому у функцію передаються копії фактичних значень

аргументів. Тобто при передаванні за значенням у стек записуються копії фактичних параметрів і оператори функції працюють з цими копіями, тому їхня зміна у функції не впливає на вихідне значення.

Параметри зі значеннями за замовчуванням

Щоб спростити виклик функції, в її заголовку можна задати значення параметрів за замовчуванням. Ці параметри мають бути останніми у списку й можуть опускатися при виклику функції.

Якщо при виклику параметр не вказаний, то мають бути опущені усі параметри, що стоять після нього. Як значення параметрів за замовчуванням можуть використовуватися константи, глобальні змінні й вирази.

Розглянемо приклади прототипів функцій з параметрами за замовчуванням.

```
int f(int a, int b = 0); // Параметр b має значення за  замовчуванням 0.
void f1(int, int = 100, char* = 0);
void f2(int x = g);       //g – глобальна змінна.
```

Варіанти виклику цих функцій:

```
f(100); // Виклик функції з першим параметром 100, другий за замовчуванням – 0
f(A, 1); // Виклик функції з першим параметром – значенням змінної A, другим – 1
f1(A); // Виклик функції з другим та третім параметрами за замовчуванням
f1(a, 10); //Виклик функції зі значенням третього параметра за замовчуванням
f1(a, 10, "Hello"); // Виклик без значень параметрів за замовчуванням
```

Рекурсія

Рекурсія – це такий спосіб організації обчислювального процесу, за якого функція звертається сама до себе. Такі звернення називаються рекурсивними викликами, а функція, що містить рекурсивні виклики, – рекурсивною. Це означає, що для вирішення певної задачі потрібно серед допоміжних підзадач вирішити таку саму задачу, тільки з іншими значеннями параметрів. Розрізняють пряму та непряму рекурсію. Пряма рекурсія полягає в тому, що певна функція безпосередньо викликає сама себе, а непряма – коли дві чи більше функцій викликають одна одну.

Рекурсія повинна мати всередині себе умову завершення, за якою наступний крок рекурсії вже не виконуватиметься, інакше вона стане нескінченною.

Створення нових копій рекурсивної функції до виходу на граничну умову називається рекурсивним спуском. Максимальна кількість копій рекурсивної функції, яке водночас може міститися у пам’яті комп’ютера, називається глибиною рекурсії. У разі відсутності граничної умови необмежене зростання кількості таких копій призведе до аварійного завершення програми за рахунок переповнення стека. Завершення роботи рекурсивних функцій, аж до самої першої, яка ініціювала рекурсивні виклики, називається рекурсивним підйомом.

Класичним прикладом рекурсивної функції є обчислення факторіала числа.

```
long Factorial(long n) {
   if (n == 0 || n == 1) return 1; //умова виходу з рекурсії
   return (n * Factorial(n - 1)); //виклик функції з її тіла
   }
```

Якщо у функції main() зустрінеться виклик функції factorail(3), при виконуванні цієї функції буде викликано функцію factorail(2), яка своєю чергою викличе factorail(1). Для останньої спрацює умова зупинки рекурсії (n==1) і у функцію factorail(2) буде

повернуто значення 1 та обчислено значення 2 (2! = 2), яке у свою чергу буде повернуто до функції factorail(3) і буде використано для обчислення 3!.

Отже, програма має 3 рекурсивні виклики. У результаті кожного рекурсивного виклику функція зберігає своє значення змінної n, оскільки до третього виклику у неї буде вже три окремих змінних з ім'ям n, при цьому кожна має власне, відмінне від інших, значення, однак тіло функції буде присутнім у пам'яті в єдиному екземплярі.

У багатьох випадках рекурсивні функції можна використовувати замість циклів. Більше того, є такі задачі, які за допомогою циклів розв'язати дуже складно або взагалі не можливо.

Завдання 1

1. Написати функцію для реалізації завдання згідно варіанту. Значення змінних потрібно вводити з клавіатури. Використання глобальних змінних у підпрограмах не допускається. Інформація у підпрограми повинна передаватися лише за допомогою параметрів. Виведення результатів роботи підпрограм повинне виконуватися в головній програмі. Введення та виведення даних необхідно супроводжувати відповідними текстовими повідомленнями.
2. Побудувати блок-схему алгоритму програми та функції.

Варіант 1

Ввести два цілих числа. Написати вираз, який приймає істинне значення, якщо остання цифра більшого числа не дорівнює 5.

Варіант 2

Ввести дійсне число X. Написати вираз, який приймає істинне значення, якщо найближче більше від X ціле число Y закінчується на 9.

Варіант 3

Написати вираз, який повертає наймолодшу цифру результату округлення більшого із двох заданих дійсних чисел до найближчого цілого.

Варіант 4

Написати вираз, який повертає суму двох наймолодших цифр числа, яке є результатом округлення до найближчого цілого меншого із двох заданих дійсних чисел.

Варіант 5

Написати вираз, який приймає значення більшої із двох найстарших цифр дробової частини дійсного числа X.

Варіант 6

Написати вираз, який визначає суму за модулем цілої та дробової частини заданого числа.

Варіант 7

Написати вираз, який повертає старшу цифру з дробової частини меншого з двох заданих дійсних чисел

Варіант 8

Записати вираз, який повертає відстань від точки (X,Y) до кола радіуса R з центром у точці (X0,Y0).

Варіант 9

Ввести ціле число X. Написати вираз, який приймає істинне значення, якщо найближче ціле до X є кратним 2 або 3.

Варіант 10

Написати вираз, який визначає найменше із трьох заданих цілих чисел

Варіант 11

Написати вираз, який повертає суму коренів квадратного рівняння, заданого коєфіцієнтами.

Варіант 12

Написати вираз, який повертає істину, якщо квадратне рівняння, задане коєфіцієнтами, має два різних дійсних кореня.

Варіант 13

Ввести ціле число. Написати вираз, який приймає істинне значення, якщо остання цифра числа кратна 3.

Варіант 14

Написати вираз, який повертає добуток цифр введеного двоцифрового числа.

Варіант 15

Ввести дійсне число X. Написати вираз, який повертає квадрат числа, якщо воно парне.

Варіант 16

Ввести дійсне число X. Написати вираз, який приймає істинне значення, якщо більше, найближче до X, ціле число є парним

Варіант 17

Ввести два цілих числа. Написати вираз, який приймає істинне значення, якщо остання цифра меншого числа дорівнює 5.

Варіант 18

Написати вираз, який міняє місцями першу та другу цифри заданого двоцифрового цілого числа.

Варіант 19

Написати вираз, який міняє місцями дробову та цілу частини заданого додатнього дійсного числа.

Варіант 20

Написати вираз, який повертає площу трикутника, заданого довжинами сторін.

Завдання 2

1. Написати підпрограму для обчислення виразу поданого як функція. Змінні, що використовуються у виразах, повинні бути дійсного типу. Значення змінних потрібно вводити з клавіатури.
2. Використання глобальних змінних у підпрограмах не допускається. Інформація у підпрограми повинна передаватися лише за допомогою параметрів. Виведення результатів роботи підпрограм повинне виконуватися в головній програмі. Введення та виведення даних необхідно супроводжувати відповідними текстовими повідомленнями.

3. Побудувати блок-схему алгоритму програми та кожної функції.

Варіант 1

$$\frac{k^2(1+p,q^2)-k(qp,1)}{1+k(p^2,q)},$$

$$\text{де} \quad k(x,y)=\frac{\sin x}{y^2}+\frac{\cos y}{x^2}$$

Варіант 2

$$\frac{h(s^2,t^2)+h^2(s+t,1)}{1+h^2(st,2)},$$

$$\text{де} \quad h(x,y)=\frac{xy}{1+x^2y^2}$$

Варіант 3

$$\frac{k(1+pq,q^2)+k^2(p,p^2)}{1+k(pq+q^2,p)},$$

де

$$k(x,y)=\frac{x}{1+\sin^2 y}+\frac{y}{1+x^2}$$

Варіант 4

$$k^2\left(p+\sqrt{q},q-\sqrt{p}\right)-k(1,p+q)$$

де

$$k(x,y)=\frac{x}{\left|x^3+y^3\right|}+\frac{y}{|x+y|}$$

Варіант 5

$$\frac{g(s^2,t+1)+g(t^2,s+1)}{1+g^2(s+t,st)},$$

де

$$g(a,b)=\frac{\sin ab}{a^2+b^2}$$

Варіант 6

$$\frac{f(1,t+s,s)+f(t,st,1)}{1+f^2(s,1,t)},$$

$$\text{де} \quad f(a,b,c)=a\sin b+b\sin a+c^2$$

Варіант 7

$$\frac{h(1,st)+h^2(t+s,1)}{1+h^3(s,t)},$$

$$\text{де} \quad h(a,b)=a^2-\sin b\cos a+b^2$$

Варіант 8

$$\frac{1+2t(1+y^2)-t^2(2y)}{10+\sqrt{t(1+|y|)}},$$

$$\text{де} \quad t(x)=\frac{x^2}{2}-\frac{x^4}{24}-1$$

Варіант 9

$$\frac{g(s,t-1)+g^2(t^2,s)}{g^2(s^2+t^2,1)},$$

$$\text{де} \quad g(x,y)=\frac{x}{y^2}-\sin\frac{y}{x^2}$$

Варіант 10

$$\frac{g(1,s)+(1+g^2(t,1))^2}{1+g^3(s+t,1)},$$

$$\text{де} \quad g(a,b)=a^2+ab+b^2$$

Варіант 11

$$\frac{h(1,st)+h^2(t+s,1)}{1+h^3(s,t)},$$

де

$$h(a,b)=a^2-\sin b\cos a+b^2$$

Варіант 12

$$\frac{h^3(t^2,1)+h(1,ts^2)}{1+h^2(s,t)},$$

де

$$h(a,b)=a^2\sin b+b^2\cos a$$

Варіант 13

$$\frac{h(s^2,1+t)+h(1,st)}{1+h^2(s,t)},$$

де

$$h(a,b)=\frac{a+b+a^2b^2}{a^2+b^2}$$

Варіант 14

$$\frac{h(a,b,1)+h(1,a,b)}{1+h(a^2+b^2,1,0)},$$

де

$$h(x,y,z)=\frac{x+y+z}{x^2+y^2+z^2}$$

Варіант 15

$$\frac{g(x^2,1,y)-g(y^2,x,1)}{1+g(\sqrt{x},y,1)},$$

де

$$g(a,b,c)=\frac{a^2+\sin b+1}{1+c^2}$$

Варіант 16

$$\frac{g(2,s)+(1+g^2(t,1))^3}{\sqrt{1+g^2(s,t)}},$$

де

$$g(a,b)=\frac{ab}{a^2+b^2}$$

Варіант 17

$$\frac{f(3)+f(x+1)+1}{1-f^2(y+1)},$$

де

$$f(a)=\frac{a^2+1}{\sin^2 a+1}$$

Варіант 18

$$\frac{f(x)+f(1+f(x))}{1+f^2(1+f^2(x))},$$

де

$$f(a)=\sin^2 a+a^2+1$$

Варіант 19

$$\frac{p^2(1+s^2)+p^3(1-s^3)}{1+p(10s)},$$

де

$$p(x)=1+\sin^2 x$$

Варіант 20

$$\frac{f^2(x)+2f(1+x+x^2)}{1+f^3(\frac{x}{2})},$$

де

$$f(a)=\frac{\sin a+\cos a}{1+a^2}$$