**Подпрограмма**

В жизни мы часто алгоритмы разделяем на части.

Например, алгоритм "Сходить в магазин за хлебом":

* 1. Дойти до магазина
  2. Купить хлеб
  3. Вернуться домой

Если вспомнить, то "дойти до магазина" состоит из более мелких действий:

* 1. Одеться
  2. Обуться
  3. Открыть дверь
  4. Выйти из дома
  5. Закрыть дверь
  6. Выйти на улицу
  7. Обойти здание
  8. Войти в дверь с вывеской "Магазин"

В этом случае, на самом деле мы объединили несколько небольших действий в одно, и дали ему наименование.

В программировании тоже соблюдается этот подход. Отдельные небольшие алгоритмы, выполняющие часть поставленной задачи, реализованные в программном коде и имеющие собственное наименование, называются **подпрограммами**.

Давайте вспомним алгоритм вычисления расстояния между двумя точками на плоскости:

**1.** input($x1, $y1);

**2.** input($x2, $y2);

**3.** $dx = abs($x2 - $x1);

**4.** $dy = abs($y2 - $y1);

**5.** $distance = sqrt(sqr($dx) + sqr($dy));

**6.** output($distance);

Разбор псевдокода

Обратите внимание, что мы ввели:

* + управляющие конструкции:
    - * присваивание мы обозначили через знак "=";
      * ввод данных обозначили словом input;
      * вывод данных обозначили словом output;
      * переменные выделили знаком $;
  + операторы суммы (+) и разности (-);
  + функции:
    - * *abs* — абсолютная величина (модуль);
      * *sqr* — квадрат числа;
      * *sqrt*— квадратный корень.

Что такое*abs*? Это отдельная функция, которая вычисляет абсолютную величину числа. Она существенно упрощает восприятие алгоритма и немного уменьшает количество программного кода (вместо условия — одна функция *abs*).

Если такой функции в нашем языке программирования нет, то нам нужно ее создать.

Если вспомнить математику — абсолютная величина равна противоположному значению числа, если число меньше нуля, и самому числу, если число больше или равно нулю.

Это можно выразить в коде:

**1.** **if** ($a < **0**) {

**2.** $abs = -$a;

**3.** } **else** {

**4.** $abs = $a;

**5.** };

Разбор псевдокода

Обратите внимание, что мы ввели:

* + управляющие конструкции:
    - * + присваивание мы обозначили через знак "=";
        + переменные выделили знаком $;
        + условие обозначили как *if* (условие) {действие при выполнении условия}*else* { действие при невыполнении условия};
    - оператор изменения числа на противоположное (-).

Но чтобы нам можно было использовать этот код в другом участке программного кода, нам нужно оформить его как подпрограмму:

**1.** **function** **abs**($arg) {

**2.** **if** ($arg < **0**) {

**3.** **return** -$arg;

**4.** } **else** {

**5.** **return** $arg;

**6.** };

**7.** };

Разбор псевдокода

Обратите внимание, что мы ввели:

* + управляющие конструкции:
    - * переменные выделили знаком $;
      * условие обозначили как*if* (условие) {действие при выполнении условия} *else* { действие при невыполнении условия};
      * подпрограмму обозначили через *function*, а возврат результата — через *return*;
  + оператор изменения числа на противоположное (-).

Переменная $arg в нашем случае является **аргументом** функции.

**Аргументы** — это данные, которые требуются для выполнения подпрограммы и передаются программой, которая вызвала подпрограмму. То есть аргументы передаются в подпрограмму при ее вызове.

Через return **возвращаем** значение, которое подпрограмма вернёт в вызвавшую ее программу.

**Важное замечание!** Не путайте возврат значения и вывод! Это частая ошибка новичков. ВОЗВРАТ производится подпрограммой в родительскую функцию. ВЫВОД данных - сообщение их пользователю.

Использование подпрограмм даёт сразу несколько преимуществ:

* 1. Единожды написанную в коде функцию можно использовать многократно;
  2. При использовании программ можно структурировать код, сделав его более читаемым.

Давайте предположим, что функции для нахождения квадрата числа в языке программирования тоже нет. В этом случае мы можем написать свою:

**1.** **function** **sqr**($arg) {

**2.** **return** $arg \* $arg;

**3.** };

Разбор псевдокода

Обратите внимание, что мы ввели:

* + управляющие конструкции:
    - * переменные выделили знаком $;
      * подпрограмму обозначили через *function*, а возврат результата — через *return*;
  + оператор произведения (\*).

Теперь запишем всё, что получилось:

**01.** **function** **abs**($arg) {

**02.** **if** ($arg < **0**) {

**03.** **return** -$arg;

**04.** } **else** {

**05.** **return** $arg;

**06.** };

**07.** };

**08.**

**09.** **function** **sqr**($arg) {

**10.** **return** $arg \* $arg;

**11.** };

**12.**

**13.** input($x1, $y1);

**14.** input($x2, $y2);

**15.** $dx = abs($x2 - $x1);

**16.** $dy = abs($y2 - $y1);

**17.** $distance = sqrt(sqr($dx) + sqr($dy));

**18.** output($distance);

Разбор псевдокода

Обратите внимание, что мы ввели:

* + управляющие конструкции:
    - * присваивание мы обозначили через знак "=";
      * ввод данных обозначили словом input;
      * вывод данных обозначили словом output;
      * переменные выделили знаком $;
      * условие обозначили как *if* (условие) {действие при выполнении условия} *else* { действие при невыполнении условия};
      * подпрограмму обозначили через*function*, а возврат результата — через *return*;
  + операторы суммы (+) и разности (-);
  + функции:
    - * *sqrt* — квадратный корень.

Подпрограммы выполняются только если запустить их из основного цикла.

Функцию нахождения расстояния тоже можно вынести отдельно. Это позволит ее переиспользовать:

**01.** **function** **abs**($arg) {

**02.** **if** ($arg < **0**) {

**03.** **return** -$arg;

**04.** } **else** {

**05.** **return** $arg;

**06.** };

**07.** };

**08.**

**09.** **function** **sqr**($arg) {

**10.** **return** $arg \* $arg;

**11.** };

**12.**

**13.** **function** **distance**($x1, $y1, $x2, $y2) {

**14.** $dx = abs($x2 - $x1);

**15.** $dy = abs($y2 - $y1);

**16.** **return** sqrt(sqr($dx) + sqr($dy));

**17.** };

**18.**

**19.** input($x1, $y1);

**20.** input($x2, $y2);

**21.** output(distance($x1, $y1, $x2, $y2));

Разбор псевдокода

Обратите внимание, что мы ввели:

* + управляющие конструкции:
    - * присваивание мы обозначили через знак "=";
      * ввод данных обозначили словом input;
      * вывод данных обозначили словом output;
      * переменные выделили знаком $;
      * условие обозначили как *if* (условие) {действие при выполнении условия} *else* { действие при невыполнении условия};
      * подпрограмму обозначили через *function*, а возврат результата — через *return*;
  + операторы суммы (+) и разности (-);
  + функции:
    - * *sqrt* — квадратный корень.