%hash

У цій статті описані основні функції та правила синтаксису для роботи з хешами в Perl.

Хеш - це асоціативний масив, оскільки звернення до його значень здійснюється за допомогою скалярних ключів, а не його числових індексів. В інших мовах програмування хеші називають інакше, наприклад, просо асоціативними масивами, словниками, списками або картами.

Хеш — це невпорядкована група пар ключ-значення. Ключі є унікальними струнами. Значення є скалярними значеннями. Кожне значення може бути числом, рядком або посиланням. Кожен хеш-ключ пов’язаний з одним значенням , і всі ключі є унікальними всередині однієї хеш-структури. Це означає, що повторювані ключі заборонені.

Хеші, як і інші змінні Perl, оголошуються за допомогою ключового слова **my**. Назві змінної передує знак відсотка (**%**). Це невеликий мнемонічний трюк, який допоможе вам нагадати про структуру ключ-значення.

Хеші є одним з основних типів даних Perl.

Хеші поділяють на однорівневі та багаторівневі. Також можуть зберігатись в своїй формі (%hash) або в формі посилання ($hashref).

Хеш є найкращим типом даних для комунікації між складними функціями, коли функція може повертати різний тип даних або їх кількість.

Також, при такому підході, вам буде легко вдосконалювати ваші програми з можливістю збереження функціонування між різними версіями.

Оголошення, ініціалізація та очищення

Оголошення (декларування)

виконується без присвоєння значень, наступним чином:

```
my %hash;
```

зазвичай це використовують для того, щоб відкрити змінну в певній області видимості, коли присвоєння виконується в різнорівневих блоках.

Очищення

Якщо в процесі виконання програми, вам потрібно очистити або знищити хеш, робіть це так:

```
%hash = ();
```

або так:

```
%hash = undef;
```

Насправді, оголошену змінну не можливо видалити, а лише очистити для звільнення оперативної пам'яті або для повторного використання з новими даними.

Ініціалізація (присвоєння)

Ініціалізація це присвоєння пустій змінній якогось значення:

```
%hash = ();
```

Оголошення та ініціалізацію можна об'єднати в одну команду, зразу присвоївши значення:

```
my %hash = ();
```

Подібно до синтаксису для масивів, хеші також можуть бути оголошені за допомогою списку значень, розділених комами:

```
my %hash = ('monday', 65, 'tuesday', 68, 'wednesday', 71, 'thursday', 53, 'friday', 60);
```

У наведеному вище коді Perl приймає перший запис у списку як ключ («monday»), а другий запис — як значення цього ключа (65). Третій запис у списку («tuesday») буде оголошено як ключ, а четвертий запис (68) як його значення тощо.

Оператор «**жирна кома**» виглядає як стрілка ('⇒') і дозволяє оголошувати пари значень ключів замість коми. Це робить код більш чистим і читабельним. Крім того, під час використання жирної коми не потрібно брати рядки в лапки для ключів. Використовуючи жирну кому, те саме оголошення %hash виглядатиме так:

```
my %hash = (
    monday    => 65,
    tuesday   => 68,
    wednesday => 71,
    thursday  => 53,
    friday    => 60,
);
```

Все вищезгадане також підходить для форми посилання ($hahref), замінивши “%” на “$” та прості дужки “()” на фігурні “{}”.

Наповнення, зміна та видалення

- ключем може бути будь який скаляр.
- ключ може бути переданий в змінній.
- якщо ключ має в собі пробіл або назву оператора - його необхідно взяти в лапки.

- числове значення треба передавати без лапок.
- рядкова значення треба передавати в лапок.
- результат обрахунку - в простих жужках.
- результат функції - виклик функції.

При зверненні до елементу хешу по ключу в контексті хешу (%hash), необхідно замінити “%” на “$” та у фігурні дужки “{}” вказати ключ:

```
$hash{key} = 'value';
```

При зверненні до елементу посилання по ключу в контексті посилання ($hashref), необхідно після назви посилання вставити '→' та у фігурні дужки “{}” вказати ключ:

```
$hashref->{key} = 'value';
```

так відбувається присвоєння та зміна значень за ключем.

За такими ж правилами відбувається видалення елементів по ключу, вказавши попереду оператор **delete** та забравши присвоєння:

```
delete $hash{key};
```

```
delete $hashref->{key};
```

Також при видаленні можна зберігати значення в іншу змінну, наприклад в скаляр:

```
my $a = delete $hash{key};
```

```
my $a = delete $hashref->{key};
```

Таким чином в змінну $a буде збережено значення, яке було видалено з хешу (посилання).