Езика за програмиране Python

Михаил Михайлов Здравков

Python - е интерпретируем, интерактивен, обектно-ориентиран език за програмиране, създаден от Гуидо ван Росум в началото на 90-те години.

I. **Въведение**.

Python е език от високо ниво, който често бива сравняван с Perl, Java и Ruby. Езикът е интуитивен и удобен. Програмите написани на него са компактни и четими.

II. **Принципи**.

Python е до голяма степен минималистичен език, лишен от условности като: стандартните скоби около условието на if-statement или на фигурните скоби за начало и край на даден блок, както и това, че липсва ‘;’ в края на всеки оператор. Групирането на изразите става чрез отстъп. Тези му особености дават бързина на писане и лесна четливост на програмите. Това прави Python изключително удобен език за писане на малки програми, при които просто ти трябва крайния резултат. Програмите на Python се делят на модули, които могат да бъдат използвани отново и в други програми.

Тъй като Python е език, който се интерпретира, се спестява значително време за разработка, тъй като не са необходими компилиране и свързване (*linking*) за тестването на дадено приложение. Освен това, бидейки интерпретируем език с идеология сходна с тази на Java, приложение, написано на него, е сравнително лесно преносимо на множеството от останали платформи (или операционни системи).

Вместо цялта функционалност на езика да бъде вградена в ядрото му, Python е създаден доста изключително гъвкав и разширим. Нови вградени модули могат лесно да бъдат написани на C, C++ или CPython. Този дизайн с малко ядро и голяма стандартна библиотека и лесно разширим интерпретатор е бил замислен от Ван Росум още от самото начало, поради неговото раздразнение към ABC(където съществува точно обратната представа – голямо ядро и малка стандартна библиотека). Дизайнът на Python предлага ограничена поддръжка за програмиране с функции в традицията на Lisp. Създателите на Python промотират, че създават „култура” или „идеология”, базирана на това какъв искат езика да бъде – семпъл и красив.

Михаил Михайлов Здравков – Технологично училище “Електронни системи” към ТУ – София
www.elsys-bg.org
mihail0zdravkov@gmail.com

III. **Интерпретатора на Python.**

Интерпретатора на Python е програма, чрез която се компилират и изпълняват програми написани на Python. Интересното на интерпретатора е, че можете да изпълните отделни парчета код, без никакво компилиране, ако да кажем, искате да видите просто дали това парче работи, как работи и какво ще стане при изпълнението му. В интерпретатора може да извикване командата help() за всяка вградена функция или библиотека, което ще изведе информация за нея. Например, ако в интерпретатора въведете:

```
print ‘Hello, world!’
```

То изхода ще бъде:

```
Hello, world!
```

Интерпретатора на Python може дори да бъде използван като калкулатор.

IV. **Променливи**.

Променливите в Python не се нуждаят от деклариране. Една променлива се създава, когато й дадеш стойност. Например:

```
a = 13;
```

Когато просто дадем стойност на дадена променлива, тя автоматична се създава. Не е нужно никъде да се оказва и тип на променливата. Променливата в Python е нещо като просто указател, който сочи към място в паметта. Ако сега се изведе стойността на променливата ‘a’, то тя ще има стойност 13 и ще бъде от тип инт. Но ако се даде нова стойност на ‘a’, която е съвсем друг тип, например:

```
a = ‘Hello’
```

То променливата ‘a’ е вече от тип стринг и нейната стойност е „Hello”. В случая, до различни символи от ‘a’ можем да достигаме, като използваме a[index].

V. Внимавайте за грешки

В езика Python дадено парче код се проверява за грешки, общо взето, едва когато се стигне до самото парче код, в изпълнението на програмата. Това е пример за това как един плюс от една страна е минус от друга. Защото в такива случаи, може да се случи да намерите грешка едва при доста дълго тестване на програмата. Ето пример:

```
if statement :
    func1()
else :
    func2()
```

В горните няколко реда код имаме един съвсем прост условен оператор, в който викаме две функции. Ако func2() е функция, която не съществува, то ние ще получим съответната грешка, едва когато кода е в изпълнение и условието на условния оператор е грешно. В Python често времето, което печелим заради удобното и бързо писане на програмата, можем да загубим в тестване на софтуера.

Точно поради тези проблеми със сигурността се водят множество спорове дали Python е подходящ за разработване на големи софтуерни продукти. Независимо от това, Python все пак е един от водещите езици днес и големи компании, като Google например признават, че ползват Python много често, когато трябва да свършат някоя задача бързо, когато е важен крайния резултат да бъде получен бързо.

VI. Вградени модули

В Python има множество, разнообразни вградени модули, които да улеснят писането на програми за вас. В днешно време модерната практика е преди да се направи дадена задача, да се провери дали някой не я направил преди вас. Така печелите време и усилия. Python е много полезен в това отношение, тъй като много от стандартните задачи, които може да ви се наложи да решите, вече са вградени. Освен това интерпретатора лесно ви помага да разучавате вградени модули и техните елементи с функцийте help() и map() например. Включването на външни модули в програма се извършва с извикването на оператора:

```
import module
```

VII. Обектно ориентирано програмиране и Python

Обектите и Обектно ориентираното програмиране са в сърцето на начина, по който Python работи. Езика позволява възможност и за писане на програми, без разбиране на Обектно ориентираното програмиране, но знанието, как то работи е задължително за да станеш нещо повече от начинаещ в Python.

VIII. Python и другите езици

Python е повлиян от езици като ABC, C/C++, Java, Lisp, Perl и тн. Затова и има много прилики с тези езици. Повечето от стандартните му оператори са идентични с тези на C и C++, различавайки се с оператора **, който повдига на степен. Например:

```
a = 10
b = 2
c = a**b
print c
```

Резултата ще бъде:

```
100
```

Оператора ** повдига стойността отляво със степента отдясно.

Python, както и много други динамични езици за програмиране, се използва и като скриптов език.

Python е повлиял на много езици, като:
Boo, Cobra, JavaScript, Ruby и др. Създателя на Ruby Юкихиро Мацумото казва: „Исках скриптов език, който да е по-мощен от Perl и по-обектно-ориентиран от Python. Заради това аз реших да създам мой собствен език.”.

Програмите написани на Python са обикновено по-бавни от тези на Java или C++, но са 3-5 пъти по-кратки от еквивалента си написан на Java и 5-10 пъти по-кратки от този написан в C++. Съществува твърдението, че това, което един Python програмист може да направи за

два месеца, ще бъде свършено от двама C++ програмисти за една година. Тази разлика може би е резултат от вградените сложни типове в Python, както и от динамичното му писане. За пример, един Python програмист не губи време в деклариране на типовете на променливи, а мощния лист на Python и речниковите му типове, за които е вградена в самия език богата поддръжка, намират приложение в почти всяка Python програма. Python също така е перфектен за използване като лепило между различни компоненти написани на C++. Езиците Lisp и Scheme са близки до Python с тяхната динамична семантика, но толкова различни в подхода си към синтаксиса, че сравнението най-често става религиозен спор. Дали липсата на синтаксис в Lisp е плюс или минус. Обикновения Lisp е голям (във всеки смисъл) и света на Scheme е фрагментиран между множество, несходни помежду си версии. Докато Python има една единствена, безплатна, компактна имплементация.

IX. CPython и други имплементации

Python има множество имплементации, като: CPython, IronPython, Jython(Java имплементация), PyPy и др. Както и различни диалекти (Cython, RPython, Stackless Python). Най-известната му, обаче, и широко използвана е CPython, който е написан на C. CPython е байткодов интерпретатор със няколко езика, включително и C.

X. История

Python е създаден в края на осемдесетте и неговата имплементация започва през Декември 1989 година от Гуйдо ван Росум в CWI в Холандия, като наследник на езика ABC. Ван Росум е принципен автор на Python и продължаването на централната му роля в решаването на развитието на Python е отразено и в титлата дадена му от общността на Python, Доброжелателен Диктатор за Живот. Python 2.0 излиза на 16 Октомври 2000 година, с много нови големи подобрения, включително garbage collector (боклукчия?) и поддръжка за Unicode. Най-важната промяна, обаче, е самият процес на създаване и преминаването му към по-прозрачен и създаван-от-общността процес. Python 3.0 (също наречен Python 3000 или py3k) става голямо събитие с излизането си на 3 Декември 2008 година, след дълъг период на тестване. Много от новостите му са прехвърлени и на по-старите версии 2.6 и 2.7. Python е награждаван за TIOBE Програмен език на годината два пъти (през 2007 и 2010), награда, която е давана на програмния език с най-голям прираст в популярността през последната година, измерена от TIOBE.

XI. Заключение

Python е изключително динамичен, гъвкав и интуитивен език. Близостта му до обикновения път на мисълта го прави удобен за ползване, а писането на програми на него става бързо и лесно. Голяма част от стандартните функции и задачи, с които програмиста се сблъсква всеки ден, са вече решени и готови да бъдат ползвани, което спестява време и усилия. Модулността на езика позволява, това което сте писали за една програма сега, без проблеми да се използва и в друг, по-нататъшен проект. Езика разполага с множество сложни типове.

Python се интегрира лесно на различни платформи и операционни системи.

Python е един чудесен избор на език, както за начинаещи програмисти, така и много полезен инструмент за напреднали софтуерни разработчици.