Informatyka w Edukacji, Toruń, 5-7 lipca 2013

PYTHON DLA WSZYSTKICH

Jochemczyk Wanda, Olędzka Katarzyna
Ośrodek Edukacji Informatycznej i Zastosowań Komputerów w Warszawie
{wanda, katarzyna}@oeiizk.waw.pl; http://konkursy.oeiizk.edu.pl

Abstract: The Python it is a high-level programming language. Its construction allows to take the first steps in programming and create the more advanced projects as well. In this article we will present some ideas of using the Python in the teaching of school children. At first we will discuss the methods of creating the drawings by using sequences of commands, loops and functions. Second issue will present exemplary tasks for more mature students, such as the problems which need the understanding of algorithms and the data structures for calculating, processing string and the lists.

1. Wstęp

Na początku warto omówić tytuł. Python – język programowania wysokiego poziomu, często używany jako język skryptowy. Cechuje go przejrzysta i zwięzła składnia. Ma rozbudowany pakiet bibliotek, co powoduje, że znajduje liczne zastosowania zarówno edukacyjne jak i komercyjne.

Rysunek 1. Logo

Dlaczego można zaryzykować stwierdzenie „dla wszystkich”? Po pierwsze, jest rozwijany jako projekt Open Source, jest darmowy, a jego interpretery są dostępne na wiele systemów operacyjnych – Windows, GNU/Linux, MacOS, … Jest to bardzo ważna cecha w edukacji, w ten sposób nie promujemy komercyjnych produktów, ale wprowadzamy uczniów w świat wolnego oprogramowania, umożliwiając pracę na w różnych systemach operacyjnych. Po drugie, nie wymusza jednego stylu programowania, pozwalając na różne podejścia: strukturalne, obiektowe i funkcyjne – kolejny przyjazny aspekt edukacyjny. Po trzecie, mogą w nim programować zarówno początkujący programiści, nawet uczniowie szkoły podstawowej, jak i osoby zawodowo zajmujące się programowaniem. Język ten jest powszechnie wykorzystywany w edukacji, jak i w rzeczywistych systemach informatycznych. Temu aspektowi – dostępności i przystępności języka Python poświęcimy dalszą część artykułu.

2. Grafika żółwia

Nasze wieloletnie doświadczenia w wykorzystaniu w edukacji języka Logo prowadzą nas do postawienia pytania: *Co takiego szczególnego znajduje się w filozofii Logo, że można uczyć programować nawet małe dzieci*? Można wymienić tu trzy aspekty: proste i konkretne, a przez to zrozumiałe komendy, zbliżone do języka naturalnego – idź naprzód o … kroków , obróć się o kąt …, podnieś pisak itp. Poza tym postać żółwia, graficzny symbol, który pozwala dziecku wyobrazić sobie wykonawcę algorytmu zapisanego w języku programowania. Po trzecie, równie ważne jak dwa poprzednie aspekty, semantyka operacyjna. Uczeń widzi efekty działania swoich programów. Interpretacja kodu powoduje efekt na ekranie, który względnie łatwo ocenić pod względem zgodności ze wzorcem. Uczeń widzi, jak żółw rysuje, czy wykonuje zadanie prawidłowo i może znaleźć ewentualny błąd. Mamy nadzieję, że rozszerzając ideę Seymoura Paperta programowania grafiki z wykorzystaniem żółwia, zachęcimy do zabawy ze środowiskiem Turtle w Pythonie.

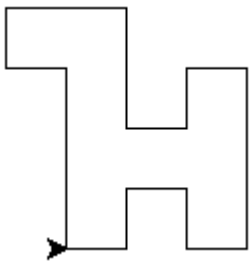
Rysunek 2. Postać żółwia modułu turtle

3. Początki

Do rysowania wykorzystujemy polecenia:

- `fd(a)` – przesuwa żółwia o podaną liczbę kroków, w zależności od stanu pisaka żółw rysuje kreskę (pisak opuszczony) lub nie rysuje (pisak podniesiony),
- `rt(kąt)` – obraca żółwia w prawo o podany kąt,
- `lt(kąt)` – obraca żółwia w lewo o podany kąt.

Spróbujmy napisać ciąg poleceń do narysowania psa, takiego jak na rysunku. Załóżmy, że jego wysokość wynosi 120.



Rysunek 3. Rysunek psa

Kod dla takiego rysunku przedstawia się następująco:

```
from turtle import *

a=120/4
#rysowanie przedniej nogi
fd(a); lt(90)
fd(a); rt(90)
fd(a); rt(90)

#rysowanie tylniej nogi
fd(a); lt(90)
fd(a); lt(90)

#rysowanie tyłu
fd(3*a); lt(90)

#rysowanie ogona
fd(a); lt(90)
fd(a); rt(90)

#rysowanie grzbietu
fd(a); rt(90)

#rysowanie głowy
fd(2*a); lt(90)
fd(2*a); lt(90)
fd(a); lt(90)
fd(a); rt(90)

#rysowanie przodu
fd(3*a); lt(90)
```

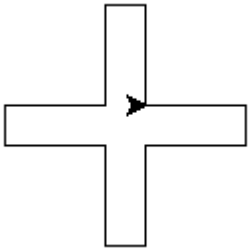
Do powtarzania poleceń można użyć instrukcji **for**. Efektywne stosowanie pętli skraca długość kodu, zwykle też staje się on bardziej czytelny.

```
from turtle import *

for i in range(10):
    #poniższe polecenia będą wykonane 10 razy
    dot(5); pu(); fd(20); pd()
```

Rysunek 4. Dziesięć kropek

W Pythonie bloki kodu zaznaczamy stosując wcięcia. W powyższym przykładzie wewnątrz instrukcji są cztery polecenia zapisane w jednej linii, w przykładzie poniżej 3 linie kodu.

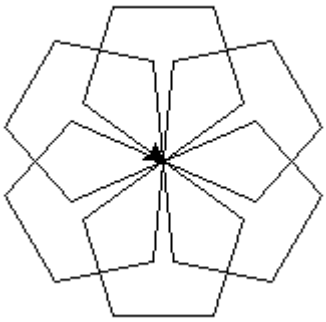
```
from turtle import *

for i in range(4):
    fd(50); rt(90)
    fd(20); rt(90)
    fd(50); lt(90)
```



Rysunek 5. Krzyżyk

Do narysowania kwiatka będziemy wykorzystywać zagnieżdżoną pętlę. Warto zwrócić uwagę na wcięcia, gdyż one informują o strukturze bloków.

```
from turtle import *

rt(36)
for i in range(6):
    for j in range(5):
        fd(50)
        rt(360/5)
    rt(60)
```



Rysunek 6. Efekt wykonania zagnieżdżonej pętli – kwiatek

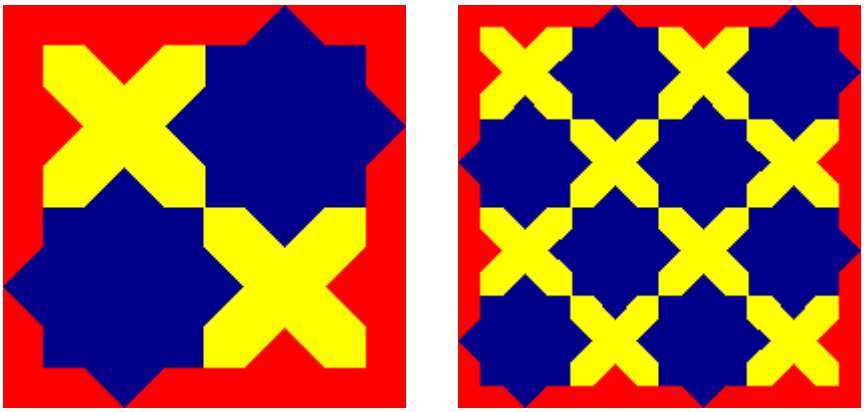
W ten sposób na prostych przykładach pokazujemy jak działa sekwencja poleceń. Uczeń poznaje, że każde polecenie ma określoną składnię. W języku Python ważna jest wielkość liter. Pole-

cenia języka jak i komendy dla żółwia wpisujemy w języku angielskim, co może być traktowane jako zaleta lub wada. Warto zwrócić uwagę, że nauka programowania dla najmłodszych obejmuje już takie zagadnienia jak podział problemu na podproblemy i co się z tym łączy pisanie funkcji, stosowanie zmiennych. Ponadto obejmuje znajdowanie elementów powtarzających się – wykorzystywanie iteracji bądź rekurencji. Do tego można dołączyć całą sferę związaną z projektowaniem, generowaniem różnych dróg rozwiązania problemu, wyborem najlepszej, kodowanie, testowanie i wyszukiwanie błędów. Stanowi to dużą garść problemów informatycznych, które można przedstawić w sposób zrozumiały dla dziecka.

Zadania konkursowe w Pythonie

Po opanowaniu rysowania prostych rysunków z wykorzystaniem iteracji warto jest rozwiązać trochę zadań z konkursów informatycznych Logo. Jednym z takich zadań jest zadanie POSADZKA z II etapu konkursu miniLOGIA 02 dla uczniów szkół podstawowych.

Posadzka jest kwadratem, który składa się z dwóch rodzajów elementów. Kolory zamalowanych elementów, to żółty i niebieski, ramka powinna być koloru czerwonego, tak jak widać na rysunkach. Napisz procedurę posadzka(n), po wywołaniu której będą rysowane takie posadzki. Parametr n jest liczbą naturalną (może przyjmować wartości od 1 do 10). Rysunek powinien być na środku ekranu. Bok dużego kwadratu powinien dla każdej wartości parametru n wynosić 300.



Rysunek 7. Posadzki dla parametrów 1 i 2

Rozwiązując to zadanie definiujemy kilka funkcji pomocniczych, z których korzystamy w głównej funkcji rysowania posadzki.

```
from turtle import *
from math import sqrt

#pomocnicza funkcja przemieszczania żółwia po ekranie
def skok (x,y=0):
    pu()
    fd(x); lt(90); fd(y); rt(90)
    pd()
```

```
#rysowanie kwadratu
def kwadrat(bok):
    color("red")
    begin_fill()
    for i in range(4):
        fd(bok); lt(90)
    end_fill()

#rysowanie niebieskiego elementu
def gwiazdka(a):
    color("darkblue")
    begin_fill()
    for i in range(4):
        fd(a); rt(45)
        fd(a*sqrt(2)); lt(90)
        fd(a*sqrt(2)); rt(45)
        fd(a); lt(90)
    end_fill()

#rysowanie żółtego elementu
def iks(a):
    color("yellow")
    begin_fill()
    for i in range(4):
        fd(a); lt(45)
        fd(a*sqrt(2)); rt(90)
        fd(a*sqrt(2)); lt(45)
        fd(a); lt(90)
    end_fill()

#rysowanie jednego kafelka
def kafelek(a):
    for i in range(2):
        gwiazdka(a)
        skok(4*a)
        iks(a)
        skok(4*a,8*a); rt(180)

#funkcja główna
def posadzka(n):
    bok=300
    a=bok/(8*n+2)
    skok(-bok/2,-bok/2)
    kwadrat(bok)
    skok(a,a)
    for i in range(n):
        for i in range(n):
```

```
            kafelek(a)
            skok(8*a)
        skok(-8*n*a,8*a)

tracer(0)
posadzka(3)
update()
```

Wywołanie funkcji `posadzka(3)` powoduje dość wolne rysowanie rysunku przez żółwia. Dla dużych parametrów i złożonych rysunków czekanie na efekt końcowy jest zbyt długie, dlatego w takich przypadkach warto stosować trzylinijkowe wywołanie złożone z funkcji `tracer()` – rysownie w pamięci, właściwego wywołania funkcji `posadzka(3)` i na końcu uaktualnienie ekranu `update()`.

Standardowy zestaw poleceń pozwala na rozwiązywanie zarówno prostych jak i trudniejszych zadań. Składnia poleceń jest klarowna, a wymuszenie stosowania wcięć powoduje, że kod staje się czytelniejszy. Osoba, która programuje ma szansę wyrobić sobie dobre nawyki. Warto jednak zwrócić uwagę na niektóre niedogodności związane z językiem i środowiskiem – na przykład błędy sygnalizowane przez interpreter są opisywane w sposób mało czytelny i zrozumiały. Odnalezienie usterki wymaga dobrej znajomości środowiska. Osobom, które programowały w tradycyjnym Logo, może przeszkadzać początkowe skierowanie żółwia w prawo, a nie jak w wielu środowiskach pionowo do góry. Rozwiązaniem jest obrót żółwia w lewo na początku lub też włączenie specjalnego trybu.

4. Zadania bez grafiki

Bardziej zaawansowane zadania wymagają definiowania funkcji liczbowych, przetwarzania łańcuchów znakowych, bądź stosowania struktur danych. Poniżej prezentujemy, jak może wyglądać przeglądanie słowa znak po znaku w pętli **for**.

```
#nadawanie wartości początkowej zmiennej s
s="gra"+"fika"
#wypisanie wartości zmiennej
print(s)

#wypisanie słowa znak po znaku
for zn in s:
    print(zn)

#słowo jako tablica
for i in range(len(s)):
    print(s[i])

#wypisanie od tyłu
for i in range(1,len(s)+1):
    print(s[-i])
```

Szyfr *Gaderypoluki* to rodzaju prostego kodowania stosowanego w harcerstwie do zapisywania krótkich wiadomości tekstowych. Jest to prosty szyfr oparty na krótkim, łatwym do zapamiętania kluczu. Klucz ten zapisuje się w formie ciągu par liter, które ulegają w tym szyfrze wzajemnemu zastąpieniu: `g` zastępujemy przez `a`, `a` przez `g`, podobnie `d` przez `e`, `e` przez `d`, itd. Litery, których nie ma liście zamienników, pozostawia się w szyfrowanym słowie bez zmian. Na przykład słowo: `alamakota` po zaszyfrowaniu ma postać `gugmgiptg`. Szyfr występuje również w innych wersjach, inne stosowane klucze to: `ma-li-no-we-bu-ty`, `po-li-ty-ka-re-nu`, `mo-ty-le-cu-da-ki`. Dla tego szyfru powstało następujące zadanie:

Napisz funkcję szyfr(k,s), której wynikiem jest zaszyfrowane słowo s według reguł przedstawionych powyżej o kluczu k. Wynikiem szyfr("gaderypoluki","alamakota") jest "gugmgiptg".

Głównym problemem w rozwiązaniu jest zamiana liter. Napiszemy funkcję `zamiana`, której parametrem jest klucz `k` i znak `zn`, który szyfrujemy. Jeśli w kluczu znajdziemy znak identyczny do podanego jako parametr i będzie on występował w kluczu na parzystej pozycji (indeks 0, 2, 4, ...) to znak `zn` po zaszyfrowaniu staje się znakiem klucza na pozycji o 1 większej. W przypadku pozycji nieparzystej (indeks 1, 3, 5, ...), znak `zn` po zaszyfrowaniu staje się znakiem klucza na pozycji o 1 mniejszej.

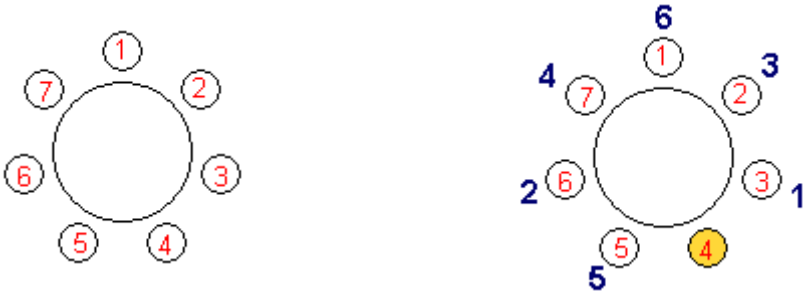
```
#zamiana znaku według klucza
def zamiana(k,zn):
    for i in range(len(k)):
        if k[i]==zn:
            if i%2==0: return k[i+1]
            else: return k[i-1]
    return zn

def szyfr(k,s):
    s2=""
    for i in range(len(s)):
        s2=s2+zamiana(k,s[i])
    return s2
```

5. Całkiem poważne problemy algorytmiczne

Python jako język wysokiego poziomu oferuje proste i złożone typy danych: listy, słowniki, zbiory, … Pozwala to na rozwiązywanie zadań algorytmicznych o mniejszym i większym stopniu trudności. Przyjrzyjmy się przykładowemu zadaniu *Okrągły stół*. Zadanie pochodzi z 3 etapu Konkursu Informatycznego dla gimnazjalistów LOGIA. Konkurs jest rokrocznie organizowany dla uczniów szkół województwa mazowieckiego. Zadanie nawiązuje do zagadnienia zwanego problemem Józefa Flawiusza, a treść jest następująca:

Przy okrągłym stole siedzi n uczestników spotkania, na krzesłach ponumerowanych od 1 do n. Kolejno co k-ta osoba wstaje i opuszcza spotkanie. Zadaniem Antka jest wskazanie osoby, która pozostanie przy stole jako ostatnia.

Pomóż Antkowi i napisz funkcję ostatni(n,k). Wynikiem funkcji jest numer krzesła zajętego przez uczestnika spotkania, który pozostanie przy stole. Parametry n i k mogą przyjmować wartości z zakresu od 1 do 100. Przykład: wynikiem ostatni(7,3) jest 4.



Rysunek 8. Osoby wstają kolejno z krzeseł o numerach: 3, 6, 2, 7, 5, 1

Nasuwającym się rozwiązaniem jest stworzenie listy wszystkich zawodników i kolejno wykreślanie wskazanego. Powstaje jednak pytanie, jak usuwać zawodników, by móc efektywnie wskazać kolejnego do wykreślenia. Jednym, choć nie jedynym, z rozwiązań jest budowanie za każdym razem nowej listy zawodników tak, by pierwszą część stanowiła lista zawodników bezpośrednio po usuniętym, a drugą część lista zawodników do tego, którego usuwamy.

Dokładniej, tak długo jak lista zawodników zawiera więcej niż jednego zawodnika wykonuj następujące kroki:

- oblicz numer zawodnika, którego trzeba usunąć (dana liczba `k` może być większa niż długość listy, wiec stosujemy operację modulo),
- zbuduj listę `pocz` od pierwszego elementu, do zawodnika usuwanego (bez tego zawodnika),
- zbuduj listę `kon` zawodników występujących bezpośrednio za usuwanym,
- połącz obie listy.

Na koniec zwróć, pierwszy i jedyny element listy.

```
def ostatni(n,k):
    #tworzenie listy od 1 do n
    lista=[]
    for i in range(1,n+1):
        lista.append(i)
    #kolejno wstaje jeden zawodnik, aż zostanie 1
    while(len(lista)>1):
        #gdy index większy od długości listy
        nr=(k-1)%len(lista)
        #tworzymy nową listę bez numeru usuniętego
        #i z początkiem na końcu
        pocz = lista[:nr]
        kon = lista[nr+1:]
        lista=kon+pocz
    return(lista[0])
```

```
print(ostatni(7,3))
print(ostatni(6,2))
```

6. Nasze spotkanie z Pythnonem

Zbliżając się do końca artykułu, pragniemy podzielić się naszym doświadczeniem uczenia się i propagowania języka Python. Nauczyciele pracowni Edukacji Informatycznej i Kształcenia na Odległość w Ośrodku Edukacji Informatycznej i Zastosowań Komputerów w Warszawie zainteresowali się językiem Python w czasie uczestnictwa w kursie e-learningowym *An Introduction to Interactive Programming in Python* prowadzonym przez profesorów Department of Computer Science Rice University w Houston (USA) na portalu www.coursera.org. Szkolenie dotyczyło tworzenia aplikacji interaktywnych w tym języku. Postanowiliśmy poszerzyć zdobytą i wiedzę i bliżej zainteresować się tym językiem.

Zaproponowaliśmy nauczycielom warsztaty rozwiązywania zadań z konkursów programistycznych organizowanych tradycyjnie w języku Logo. Zorganizowaliśmy szkolenie *Podstawy programowania w języku Python*. Dotyczyło ono głównie grafiki żółwia, ale rozwiązywaliśmy też inne problemy algorytmiczne wykorzystując przetwarzanie słów i list. Szkolenie zostało zorganizowane metodą blended learning – dwa spotkania stacjonarne i aktywności zdalne na platformie elearningowej takie jak zadania, dyskusja na forum, quiz i wspólne tworzenie bazy motywów graficznych narysowanych przez żółwia z wykorzystaniem modułu turtle.

Rysunek 9. Logo konkursu

Kolejnym krokiem było ogłoszenie ogólnopolskiego konkursu przeznaczonego dla uczniów szkół podstawowych i gimnazjum zainteresowanych informatyką i programowaniem. Polega on na samodzielnym rozwiązywaniu w języku Python zadań dotyczących grafiki żółwia. Dla uczniów gimnazjów przewidziane są także zadania nie związane z grafiką. Konkurs jest prowadzony na platformie Moodle http://konkursy.oeiizk.edu.pl. Uczestnicy zawodów mogą samodzielnie rozwiązywać zadania konkursowe w wybranym przez siebie miejscu i czasie. Swoje rozwiązania każdy z uczestników przesyła za pomocą formularza. Są dostępne materiały pomocnicze wprowadzające do programowania w języku Python. Zakończenie konkursu planowane jest na przełomie maja i czerwca. Mamy nadzieję, że będziemy mogli przedstawić wyniki i zaprezentować wnioski.

7. Zakończenie

Dostrzegając potrzebę ciągłego doskonalenia umiejętności programistycznych i zdolności myślenia algorytmicznego rozpoczęłyśmy swoją przygodę z Pythonem. Jest to okazja poszerzenia oferty edukacyjnej w naszej pracy i zainspirowania innych do twórczych poszukiwań.

Znajomy Antek wraca ze szkoły i opowiada jak to pan od informatyki pokazał mu nowe środowisko. Po chwili rozmowy okazuje się, że tata Antka, jako informatyk w swojej firmie pisze skrypty w Pythonie, które ułatwiają analizę danych na stronach internetowych. Pozostaje tylko otwarte pytanie, czy środowisko to przyjmie się w edukacji?

Literatura

1. Aplikacje demo Python Turtle http://code.google.com/p/python-turtle-demo
2. Dokumentacja modułu turtle http://docs.python.org/3.3/library/turtle.html
3. Norton P. i inni, *Python. Od podstaw*, Helion, Gliwice, 2006
4. Pilgrim M., *Zanurkuj w Pythonie*, Wikibooks, biblioteka wolnych podręczników, 2008, ostatni dostęp 10 kwietnia 2013 roku
5. Strona domowa Python Turtle http://pythonturtle.org
6. Zespół autorów, http://pl.python.org/kursy,jezyka.html
7. Zespół autorów, *Programowanie z Pythonem*, zob. w dziale Technologia Informacyjna — zajęcia, na stronie https://brain.fuw.edu.pl/edu