Czy można zaprzyjaźnić się z Pythonem?

Wanda JOCHEMCZYK, dr Katarzyna OLĘDZKA

Mimo podejmowania wielu wysiłków człowiekowi nie udaje się oswoić wszystkich zwierząt. Istnieje przekonanie, że do zwierząt, których nie sposób oswoić, należą węże. Gad ten jedynie przyzwyczaja się do obecności człowieka w swoim otoczeniu. Możemy nauczyć się dbać o węża i polubić jego obecność, ale on ze swojej natury zawsze pozostanie dzikim stworzeniem, a więc w dużym stopniu nieprzewidywalnym w swoim zachowaniu. Należy pamiętać, dla własnego dobra i dobra innych ludzi, że spotkanie z wężem może być niebezpieczne, szczególnie, gdy jest on jadowity. Nie o tym jednak będziemy rozważać. Opiszemy naszą przygodę z Pythonem – językiem programowania, którego nazwa w gruncie rzeczy nie pochodzi od zwierzęcia – pełzającego gada. Nie odwołuje się także od potwora znanego z mitologii greckiej. W latach 70. BBC emitowało serial komediowy zatytułowany „Monty Python's Flying Circus" – Latający Cyrk Monty Pythona. To właśnie do tego serialu odnosi się nazwa języka, za którego twórcę uznawany jest Guido van Rossum, holenderski programista.

Rysunek 1. Logo Pythona.

Dlaczego Python?

Istnieje wiele różnych języków programowania. Dlaczego zatem promujemy naukę właśnie języka Python? Zapytaliśmy o walory edukacyjne tego języka naszych nauczycieli – uczestników szkolenia e-learningowego dotyczącego programowania w Pythonie.

- *Jego jasna i prosta składnia pozwala skupić się na faktycznym problemie i rozwiązaniu zadania. Uczy on dobrych nawyków, m.in. organizacji kodu.* (p. Marta)

- *Czy Python? Początkowe trudności z wcięciami mijają, a po pewnym czasie są postrzegane jako ułatwienie. Właściwe też jest początkowe wprowadzanie grafiki, a następnie algorytmiki.* (p. Andrzej)

- *Python jest bardzo często wykorzystywany do tworzenia prototypów rozwiązań, bo osiąga się szybkie efekty w krótkim czasie. Jest to język o bardzo szerokim spektrum zastosowań. Pewnie ciężko dziś znaleźć dziedzinę, w której nie dałoby się go zastosować.* (p. Grzegorz)

- *Python jest językiem wymagającym bardzo precyzyjnego zapisu, ważne jest każde wcięcie, dwukropek. Przez to kod jest bardzo klarowny, jasny. Uczeń od razu uczy się struktury pętli i widzi, co w tej pętli chce zawrzeć.* (p. Katarzyna)

I jeszcze jedna wypowiedź dotycząca nauki programowania:

- *Rozwiązywanie problemów to nic innego, jak tworzenie algorytmów. Już niemowlęta tworzą algorytmy polegające na odpowiednio długim darciu dzioba czy wylewaniu łez, żeby uzyskać wzięcie na ręce, jedzenie czy suchą pieluchę na... Wówczas jeszcze nie mają świadomości, że właśnie wymyślają algorytm działania, żeby uzyskać zaplanowany efekt, czyli rozwiązać napotkany problem.* (p. Krzysztof)

Warto podkreślić, że w Pythonie mogą programować zarówno początkujący, nawet uczniowie szkoły podstawowej, jak i osoby zawodowo zajmujące się programowaniem. Ten aspekt może zachęcić uczniów do poważnego zajęcia się programowaniem. W Pythonie można prosto zapisać wiele algorytmów, a przy wykorzystaniu dostępnych bibliotek przygotować oprogramowanie dostosowane na naszych potrzeb. Ponadto sprzyja on zwięzłemu zapisowi i wymusza poprawne formatowanie kodu poprzez ścisłe zasady stosowania wcięć. Kolejną zaletą edukacyjną jest możliwość stosowania strukturalnego, obiektowego lub funkcyjnego stylu programowania. Jako projekt Open Source jest darmowy, a jego interpretery są dostępne na wiele systemów operacyjnych, m.in. Windows, GNU/Linux, MacOS. Jest to bardzo ważne w edukacji, ponieważ nie promujemy komercyjnych produktów, ale wprowadzamy uczniów w świat wolnego oprogramowania. Wkrótce także będzie można zdawać egzamin maturalny z informatyki w tym języku.

Python już w szkole podstawowej

Najmłodszych uczniów wprowadzamy w świat programowania, korzystając z języka wizualnego. Uczeń, tworząc program, nie tyle wpisuje polecenia z klawiatury, co przeciąga bloczki. Nie musi pamiętać składni poleceń, ale powinien je rozumieć, by efektywnie stosować. W zasadzie nie popełnia też błędów składniowych. Dlatego dzieci chętnie zaczynają naukę programowania od Scratcha lub międzynarodowego projektu *Godzina Kodowania*. Jednak, gdy chcemy napisać trochę dłuższy program, okazuje się, że bloczki stają się pewną przeszkodą. Kod jest długi i mało czytelny, stąd w sposób naturalny rodzi się potrzeba przejścia na tekstowy język programowania. Warto jednak pamiętać, że przejście to powinno być łagodne, a uczniom należy stawiać takie zadania, które zachęcą ich do pracy, a nie zniechęcą do nauki programowania.

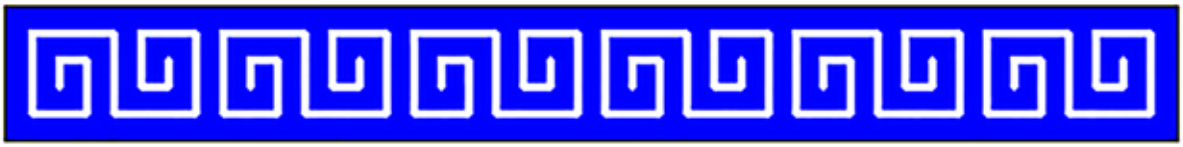
Jak rozpocząć i dlaczego? Proponujemy zacząć naukę programowania od sterowania obiektem na ekranie. Po pierwsze, stosujemy proste, a przez to zrozumiałe komendy, zbliżone do języka naturalnego – idź naprzód o daną liczbę kroków, obróć się o podany kąt, podnieś pisak itp. Po drugie, postać żółwia to graficzny symbol, który pozwala dziecku wyobrazić sobie wykonawcę algorytmu zapisanego w języku programowania. Po trzecie, równie ważna, jak dwa poprzednie aspekty, jest semantyka operacyjna. Uczeń widzi efekty działania swoich programów. Interpretacja kodu powoduje na ekranie skutek, który może ocenić pod względem zgodności ze wzorcem. Uczeń obserwuje, jak żółw rysuje, i ocenia, czy wykonuje zadanie prawidłowo. Może też znaleźć ewentualny błąd. Wszystkie te elementy znajdziemy zarówno w Scratchu – gdy rysujemy, wykorzystując pisak – jak i w projekcie *Godzina Kodowania* – sterując różnymi obiektami, w Logo, a także w Pythonie – korzystając z modułu Turtle. Rozszerzamy w ten sposób ideę Seymoura Paperta – programowania grafiki z wykorzystaniem żółwia – i zachęcamy uczniów zarówno do nauki, jak i do zabawy.

Sterowanie żółwiem

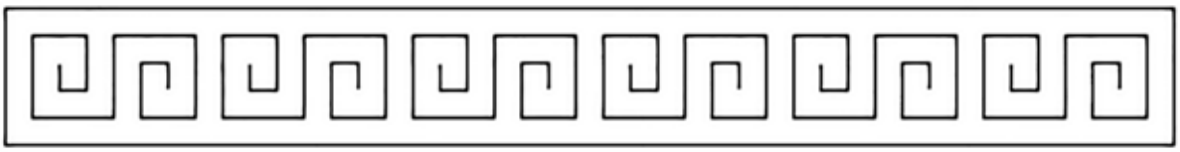
Do rysowania wykorzystujemy polecenia:

- `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,
- `pu() / pd()` – podnosi / opuszcza pisak.

Podczas tworzenia rysunków z wykorzystaniem grafiki żółwia jedną z podstawowych umiejętności jest odnajdywanie elementów powtarzających się



Rysunek 2. Motyw grecki.



Rysunek 3. Motyw grecki – rysunek konturowy.

i odpowiednie stosowanie instrukcji iteracji. O ile przy prostych przykładach uczniowie radzą sobie dobrze z zadaniem, przy bardziej skomplikowanych pojawiają się trudności. Warto zauważyć, że zagadnienie iteracji obejmuje zadania, w których występuje proste powtarzanie, powtarzanie ze zmiennym elementem oraz zagnieżdżone pętle.

Przyjrzyjmy się następującemu zadaniu. Mamy napisać funkcję, po wywołaniu której zostanie narysowany motyw grecki (rysunek 2).

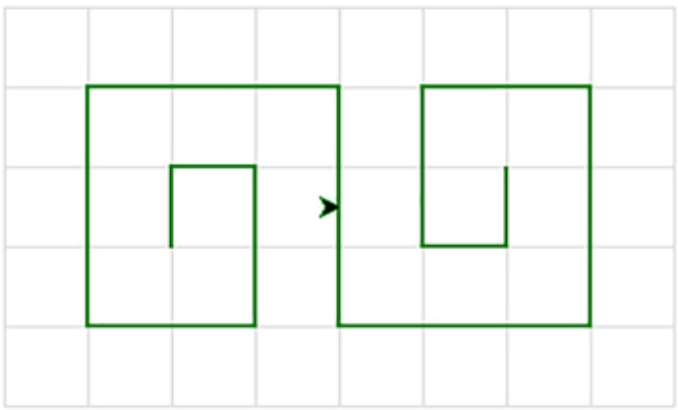
Od czego zacząć rozwiązywanie takiego zadania? Spróbujmy przedstawić rysunek motywu w sposób bardziej przejrzysty, aby widać było rysowane elementy.

Następnie możemy narysować powtarzające się elementy na pomocniczej kratce, aby dokładnie zbadać proporcje jednego elementu.

Miejsce, w którym widzimy żółwia, jest optymalnym miejscem rozpoczęcia rysowania. Żółw powinien rozpocząć i zakończyć rysowanie w tym samym miejscu. Warto stosować taką zasadę, gdyż ułatwia to łączenie elementów.

```
from turtle import *
def zawijas(a):
   for k in range(2):
      lt(90)
      fd(1.5*a);lt(90)
   for i in range(2):
      fd(3*a);lt(90)
   for i in range(2):
      fd(2*a);lt(90)
   for i in range(2):
      fd(a);lt(90)
   pu();fd(2*a);rt(90);
   bk(0.5*a);rt(90);pd()
```

Najtrudniejszy fragment rozwiązania jest poza nami. Zostało narysowanie prostokąta i powtórzenie 6 razy zdefiniowanego elementu. Motyw grecki jest rysowany białym pisakiem na niebieskim tle, musimy zatem wybrać odpowiednią kolejność rysowania, aby efekt był taki, jak na wzorcowym rysunku. Jako pierwszy narysujemy prostokąt zamalowany kolorem niebieskim, następnie po zmianie koloru pisaka na biały narysujemy 6 powtarzających się elementów.



Rysunek 4. Element powtarzający się, narysowany na pomocniczej kratce.

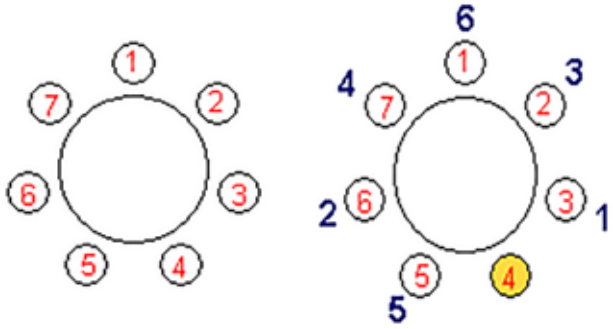
```
def motyw():
   #ustalenie wielkości
   a=10
   fillcolor("blue")
   #rysowanie prostokąta
   begin_fill()
   for i in range(2):
      fd(43*a);lt(90);fd(5*a);lt(90)
   end_fill()
   #przejście w odpowiednie miejsce
   pu();fd(4*a);lt(90)
   fd(2.5*a);rt(90);pd()
   pensize(3)
   pencolor("white")
   #rysowanie 6 elementów
   for i in range(6):
      zawijas(a)
      pu();fd(7*a);pd()
```

Rysowanie złożonego motywu trwa długo. Żółw wykonuje powoli każdy ruch. Ułatwia to naukę, ale przeszkadza w testowaniu, zajmuje za dużo czasu. Warto przyśpieszyć rysowanie, uruchamiając tworzenie motywu w pamięci i na końcu wyświetlając go na ekranie za pomocą poleceń:

```
tracer(0);motyw();update()
```

Przykład całkiem poważnego problemu algorytmicznego

Przyjrzyjmy się zadaniu „Okrągły stół". Zadanie nawiązuje do zagadnienia zwanego problemem Józefa Flawiusza, a pochodzi z 3. etapu Konkursu Informatycznego dla gimnazjalistów LOGIA13 (rok szkolny 2012/2013). Konkurs jest rokrocznie organizowany dla uczniów szkół województwa mazowieckiego (`http://logia.oeiizk.waw.pl`).



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

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**.

Nasuwającym się rozwiązaniem jest stworzenie listy wszystkich uczestników i kolejno wykreślanie osoby opuszczającej miejsce. Powstaje pytanie, jak usuwać osobę, by móc efektywnie wskazać kolejną do wykreślenia. Jednym z rozwiązań, choć nie jedynym, jest budowanie za każdym razem nowej listy uczestników tak, by pierwszą część stanowiła lista uczestników bezpośrednio po usuniętym, a drugą część lista osób do tej, którą usuwamy. Dokładniej, tak długo jak lista zawiera więcej niż jednego uczestnika, wykonuj następujące kroki:

- oblicz numer osoby, którą trzeba usunąć (dana liczba `k` może być większa niż długość listy, wiec stosujemy operację modulo),
- zbuduj listę `pocz` od pierwszego elementu, do osoby usuwanej (bez tej osoby),
- zbuduj listę `kon` osób występujących bezpośrednio za usuwaną,
- połącz obie listy `kon` i `pocz`.

Wynikiem jest 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 uczestnik,
   #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]
#przykłady wywołań
print(ostatni(7,3))
print(ostatni(6,2))
```

Na komputerze i w chmurze

W języku Python możemy pracować lokalnie na komputerze, po wcześniejszej jego instalacji. Warto dodać, że w różnych dystrybucjach Linuksa oraz MacOS-a Python jest już zainstalowany z systemem. Możemy też korzystać z serwisów, np. `http://ideone.com` lub `http://pythontutor.com`. Otwieramy przeglądarkę internetową, a następnie piszemy i uruchamiamy nasz program. W tym drugim serwisie możemy napisać skrypt, a następnie go uruchomić i wykonać krok po kroku. Dodatkowo mamy narzędzie do wizualizacji – możemy obserwować, jakie są bieżące wartości zmiennych i jak się zmieniają w czasie. Znajdziemy tam kilkanaście przykładów zaimplementowanych podstawowych algorytmów.

Warto również dodać, że mamy dostępne różne biblioteki i dodatki do tego języka. Przykładem przydatnej biblioteki dla uczniów jest *Pygame.* Jest ona przeznaczona do tworzenia gier oraz aplikacji multimedialnych. Można tworzyć w nim grafikę statystyczną i dynamiczną, odtwarzać i edytować dźwięki czy korzystać z materiałów filmowych. W sieci opublikowano wiele programów stworzonych z wykorzystaniem biblioteki *Pygame*, można je pobrać i z nich skorzystać oraz „podejrzeć", jak zostały zrobione, by się wiele nauczyć. Z kolei rysować wykresy można z wykorzystaniem biblioteki *SageMath* (*Software for Algebra and Geometry Experimentation*). Jest to dość rozbudowany pakiet, w którym obok wykresów w 2D i 3D możemy zajmować się zagadnieniami związanymi z różnego typu obliczeniami, rozwiązywaniem równań czy zagadnieniami z dziedziny kombinatoryki.

Nasze spotkanie z Pythonem

Zbliżając się do końca, pragniemy podzielić się naszym doświadczeniem uczenia się i propagowania języka Python. Zainteresowałyśmy się nim w czasie udziału w kursie e-learningowym *An Introduction to Interactive Programming in Python* prowadzonym przez profesorów z Department of Computer Science Rice University w Houston (USA) na portalu `www.coursera.org`. Szkolenie dotyczyło tworzenia aplikacji interaktywnych w tym języku. Postanowiłyśmy poszerzyć zdobytą wiedzę i bliżej zainteresować się Pythonem. Potem zaproponowałyśmy szkolenia dla nauczycieli i konkurs dla uczniów. Obecnie prowadzimy całą serię szkoleń, głównie w systemie online. Na stronie `python.oeiizk.edu.pl` są dostępne materiały pomocnicze wprowadzające do programowania w tym języku. Chociaż Python nie jest jedynym językiem, w którym można stawiać pierwsze kroki w programowaniu, jest dość specyficzny i wart poznania. Zachęcamy naszych Czytelników, o ile jeszcze tego nie uczynili, aby rozpoczęli swoją przygodę z Pythonem.

Bibliografia

1. Borowiecki M. *Python na lekcjach informatyki w szkole ponadgimnazjalnej*, IwE 2013.
2. Jochemczyk W., Olędzka K. *Python dla wszystkich*, IwE 2016.
3. Jochemczyk W., Olędzka K. *Pythonowe wyzwania dla początkujących*, IwE 2017.
4. https://code.org
5. http://python.oeiizk.edu.pl
6. http://python.org

Wanda JOCHEMCZYK i **dr Katarzyna OLĘDZKA** są nauczycielami konsultantami w Ośrodku Edukacji Informatycznej i Zastosowań Komputerów w Warszawie.