Informatyka w Edukacji, XVII
UMK Toruń, 2020

KURS JĘZYKA PYTHON 3 W FORMULE MOOC: IMPLEMENTACJA I WSTĘPNA EWALUACJA

Jakub Swacha
Katedra Informatyki w Zarządzaniu
Instytut Zarządzania
Uniwersytet Szczeciński
ul. Cukrowa 8, 71-004 Szczecin
e-mail: jakub.swacha@usz.edu.pl

Abstract. High demand for computer programming education instigates the search for forms of teaching which are effective and convenient (for both sides of this process). One of such is Massive Open Online Course (MOOC). In this work we describe an introductory course on programming in Python 3 language developed as MOOC at University of Szczecin, presenting its design assumptions, means of implementation and the results of preliminary evaluation involving e-learning experts' and students.

1. Wstęp

Szerokie zainteresowanie nauką programowania komputerów stanowi przesłankę do poszukiwania efektywnych i wygodnych dla obu stron tego procesu form nauczania. Za taką właśnie uważa się masowe otwarte kursy online (MOOC) [5]. Jeszcze przed wybuchem pandemii COVID-19 kursy tego rodzaju cieszyły się dużą popularnością ze względu na ich dostępność i stosunkowo niski koszt [15], w warunkach panującej pandemii ich znaczenie w edukacji niechybnie wzrośnie.

Język Python, który w okresie ostatnich kilkunastu lat odniósł olbrzymi sukces w praktyce i edukacji (stając się obecnie według rankingu TIOBE trzecim pod względem popularności językiem na świecie [18]), doczekał się szeregu poświęconych mu kursów MOOC. Choć są wśród nich wysokiej klasy kursy opracowane przez renomowane uczelnie (m.in. Harvard [19], Massachusetts Institute of Technology [3] i University of Michigan [13]), niestety brakuje otwartych kursów w języku polskim. Należy tu zauważyć, że choć jednym z pierwotnych znaczeń słowa „otwarte" w skrócie MOOC była ich bezpłatność dla uczestników (zob. rys. 1.2 w [6]), o tyle obecnie co raz więcej kursów dostępnych jest w formule freemium (bezpłatny kurs z płatnym certyfikatem ukończenia) lub całkowicie płatnej (dostęp do materia-

łów uzależniony jest od wniesienia opłaty). W pierwszej z wymienionych formule funkcjonuje kurs „Python Essentials" opracowany przez OpenEDG Python Institute [12], a w drugiej inne kursy dostępne w języku polskim: „Python dla początkujących" Rafała Mobilo [8], „Python. Podstawy programowania" Łukasza Pełszyńskiego [9] i „Python 3 od Podstaw do Eksperta" Arkadiusza Włodarczyka [20].

Niniejszy autor sam jest autorem jednego z pierwszych (opublikowanego w 2005 r.) polskich internetowych kursów tego języka [17]. Wspomniany kurs, mimo ulepszonej interaktywnej wersji drugiej [16], nie ma jednak charakteru kursu MOOC, jest raczej podręcznikiem elektronicznym zawierającym materiały i ćwiczenia do samodzielnego wykonania, bez harmonogramu i weryfikacji postępów w nauce, czy elementów społecznościowych, tym samym może stanowić jedynie pomoc dydaktyczną uzupełniającą naukę programowania opartą na innych formach nauczania. Podobnych do niego w charakterze udostępnionych w Internecie zasobów w języku polskim jest zresztą więcej, jak np. oparty na wiedzy społeczności StackOverflow kurs „Pierwsze kroki w języku Python" [10], czy bogaty w ciekawe przykłady, lecz niestety bardzo słabo przetłumaczony na język polski „Interaktywny samouczek w języku Python 3" [2].

Mając świadomość istniejącej luki, a jednocześnie spotykając się w codziennej pracy dydaktycznej z dużym zainteresowaniem ze strony studentów dostępnością polskiego MOOC dla języka Python, w 2018 r. niniejszy autor rozpoczął prace nad takowym kursem. Dużym bodźcem do przyspieszenia prac okazał się ogłoszony przez Narodowe Centrum Badań i Rozwoju konkurs „Kurs na MOOC" [7], umożliwiający pozyskanie środków z Programu Operacyjnego Wiedza Edukacja Rozwój 2014-2020 na realizację kursów MOOC. Dzięki uzyskanemu tą drogą dofinansowaniu możliwe było wzbogacenie treści kursu o materiały wymagające dużego jednostkowego nakładu pracy, w szczególności ćwiczenia typu „przeciągnij i upuść", a przede wszystkim filmy instruktażowe. Właśnie ten kurs jest przedmiotem niniejszego opracowania.

2. Koncepcja kursu

Za główny cel kursu przyjęto podniesienie wśród jego uczestników:

- kompetencji informatycznych, w zakresie programowania w języku Python 3 oraz samodzielnego poszerzania wiedzy na temat tego języka,
- kompetencji analitycznych, w zakresie rozumienia kodu źródłowego w języku Python 3 oraz zdolności identyfikacji elementów opisu problemu istotnych dla znalezienia rozwiązania programistycznego.

Jako że kurs powstawał w odpowiedzi na zainteresowanie ze strony studentów, to oni mieli stanowić główną kategorię jego docelowych użytkowników. Główną nie znaczy jednak jedyną: już doświadczenia z pierwszym kursem niniejszego autora,

także skierowanym pierwotnie do własnych studentów, jasno pokazały, że opublikowane w Internecie i zaindeksowane przez wyszukiwarki otwarte treści edukacyjne przyciągają zainteresowanych nimi odbiorców o bardzo różnych profilach (otrzymywano maile od: uczniów różnego poziomu szkół, hobbystów, czy też profesjonalistów na stanowiskach nieprogramistycznych, którym pojęcie podstaw języka Python potrzebne było do realizacji specyficznych zadań związanych z wykonywaną pracą). Dbając zatem o to, by materiał ujęty w kursie był wystarczający do osiągnięcia założonych w planie studiów efektów kształcenia z zakresu podstaw programowania, pamiętano jednocześnie, by nie nadać mu formy stricte akademickiej, która mogłaby być zbyt trudna w odbiorze i zbyt sztywna dla innych niż studenci grup użytkowników, w szczególności takich jak: uczniowie szkół średnich (nie tylko o profilu informatycznym), programiści zawodowi znający inne języki programowania i programiści-hobbyści. Kurs nie stawia przy tym żadnych wymagań wstępnych poza umiejętnością obsługi komputera na poziomie podstawowym (uruchamianie komputera i programów, znajomość klawiatury komputerowej i zasad edycji tekstu).

Przyjęto trójpoziomową strukturę kursu, na którą składają się:

- moduły, grupujące lekcje od zagadnień podstawowych do coraz bardziej zaawansowanych,
- lekcje, obejmujące zagadnienia z określonego zakresu tematycznego,
- jednostki, omawiające pojedyncze zagadnienia lub weryfikujące ich opanowanie.

Zdecydowano, że cały kurs składać się będzie z wprowadzenia (modułu organizacyjnego, obejmującego m.in. test otwarcia oceniający poziom wiedzy na starcie kursu), modułów głównych oraz zakończenia (obejmującego m.in. test zamknięcia oceniający poziom wiedzy na końcu kursu). Jednocześnie każdy moduł zasadniczy kursu (tj. każdy za wyjątkiem wprowadzenia i zakończenia) kończyć się będzie testem podsumowującym moduł.

Długość semestru (15 tygodni) i wymagany od studentów nakład pracy (60 godzin) wyznaczyły ramy objętości kursu. Jednocześnie mając na uwadze zarówno użytkowników nie będących studentami, a nawet studentów, lecz studiów niestacjonarnych, których harmonogram sesji zjazdowych jest często nieregularny, postanowiono unikać w kursie elementów wymuszających tempo nauki (spowalniających – np. udostępnianie treści lub zadań w określonych terminach, czy przyśpieszających – np. wymaganie ukończenie części kursu w określonych terminach) poza wytyczeniem ogólnych ram czasowych kursu (jego początku i końca). Dzięki temu, ucząc się w trybie wysoce intensywnym (po 6 godzin dziennie), cały kurs ukończyć można w zaledwie dwa tygodnie.

Kurs postanowiono opracować w dwóch wersjach językowych: polskiej (pierwotnej) i angielskiej (tłumaczenie). Potrzeba opracowania wersji angielskiej, mimo

opisywanej wcześniej dostępności wysokiej klasy otwartych kursów w tym języku, wynika m.in. z planowanego wykorzystania kursu przez grupy studenckie, w których zdarzają się obcokrajowcy, znający lepiej język angielski niż polski.

3. Zakres merytoryczny kursu

Określając zakres merytoryczny kursu zdefiniowano następujące potrzeby:

- przedstawienie w pełni składni języka, tak by osoba, która ukończyła kurs, była w stanie zrozumieć składnię dowolnego fragmentu kodu napisanego w języku Python 3,
- dogłębne wytłumaczenie działania podstawowych instrukcji języka i dostateczne wytłumaczenie działania instrukcji rzadziej używanych (tak, by z jednej strony nie przeciążać kursu zbyt dużą ilością wysoce specjalizowanych, rzadko użytecznych informacji, z drugiej zaś, by zmotywować uczestników kursu do samodzielnego poszerzania wiedzy programistycznej),
- zademonstrować wykorzystanie języka Python do rozwiązywania problemów algorytmicznych z wykorzystaniem klasycznych technik projektowania algorytmów (w nawiązaniu do przedmiotu algorytmy i struktury danych),
- zademonstrować korzystanie z niestandardowych modułów rozszerzeń, tak by osoby kończące kurs były świadome istnienia bogatej biblioteki rozszerzeń dla języka Python i sposobu korzystania z niej,
- przygotować uczestników kursu do tworzenia bardziej złożonych programów, odnosząc się do szczególnie popularnych zagadnień praktycznych występujących w tym kontekście (co jest przydatne także dla studentów, np. ze względu na stosunkowo obszerne projekty programistyczne, których wykonanie jest od nich wymagane do zaliczenia przedmiotu).

Listę ostatecznie wybranych tematów modułów zasadniczych przedstawiono w tabeli 1. Łącznie zdefiniowano 14 tematów (ostatni tydzień nauki zaplanowano na podsumowanie i przygotowanie uczestników do testu zamknięcia; nie rezerwowano osobnego tygodnia na moduł organizacyjny, uznając, że uczestnicy mogą zapoznać się z jego treścią w tym samym tygodniu co pierwszego modułu zasadniczego, przed rozpoczęciem właściwej nauki). Tematy pierwszych dziesięciu modułów odnoszą się do dwóch pierwszych potrzeb wymienionych wyżej (objaśnienie składni i zestawu instrukcji języka Python 3), moduł „Algorytmy w Pythonie” – do potrzeby trzeciej (rozwiązywanie problemów algorytmicznych), natomiast trzy ostatnie moduły – do potrzeby piątej (praktyka programowania), przy czym moduł „Korzystanie z modułów PYPI” odnosi się jednocześnie do potrzeby czwartej (używanie modułów rozszerzeń).

Tabela 1. Zakres tematyczny kursu

Moduł	**Zagadnienia**
Pierwszy kontakt z językiem Python	Jak: uruchamiać środowisko IDLE, posługiwać się nim i korzystać z dostępnej w nim pomocy online; używać języka Python do wykonywania obliczeń matematycznych; tworzyć w języku Python zmienne, używać i kasować je.
Łańcuchy znaków (napisy)	Jak: tworzyć stałe napisowe, w tym łańcuchy zawierające znaki specjalne i wartości zmiennych; wykonywać podstawowe operacje na całych napisach, w tym łączyć je i przekształcać; wycinać fragmenty napisów i operować na nich.
Programy	Jak, korzystając z języka Python i narzędzia IDLE: tworzyć proste programy wczytujące i wyświetlające tekst; tworzyć rozgałęzienia; używać asercji i wbudowanego debuggera.
Sekwencje	Jak: tworzyć sekwencje wartości dowolnego typu, zarówno modyfikowalne, jak i niemodyfikowalne; wykonywać podstawowe operacje na sekwencjach, takie jak łączenie, powielanie, przeszukiwanie i wybieranie fragmentów; tworzyć sekwencje bazujące na przedziałach liczb naturalnych.
Pętle	Jak: tworzyć pętle przetwarzające elementy sekwencji; tworzyć pętle zależne od dowolnego warunku; wykonywać wielokrotne operacje bez użycia pętli.
Zbiory i słowniki	Jak: tworzyć i modyfikować zbiory modyfikowalne i z nich korzystać; tworzyć zbiory niemodyfikowalne i z nich korzystać; tworzyć i modyfikować słowniki i z nich korzystać.
Funkcje	Podstawy definiowania i wywoływania funkcji; zasady przekazywania do funkcji parametrów i zwracania przez nie rezultatów; zasady dostępu do zmiennych lokalnych i globalnych, tworzenie i korzystanie z: funkcji zagnieżdżonych i domknięć, funkcji rekurencyjnych i funkcji anonimowych.
Programowanie obiektowe	Podstawy definiowania klas i ich składowych; mechanizm dziedziczenia, w tym przeciążanie metod i dziedziczenie po wielu klasach; definiowanie i korzystanie z metod klasowych i statycznych oraz z iteratorów i generatorów.
Moduły standardowe Pythona – przegląd	Jak: korzystać z zewnętrznych modułów; używać pseudolosowych liczb i sekwencji; korzystać z funkcji matematycznych, w tym trygonometrycznych; mierzyć upływający czas,

Moduł	Zagadnienia
	uzyskać aktualny czas i datę oraz formatować wyświetlany czas i datę; wychodzić w dowolnym miejscu z programu.
Przetwarzanie danych	Jak: zabezpieczać działanie programu przed błędnymi danymi wejściowymi poprzez obsługę wyjątków; używać funkcji dedykowanych do przetwarzania sekwencji; tworzyć listy składane, wyrażenia generujące i słowniki składane; przetwarzać tekst z wykorzystaniem wyrażeń regularnych.
Algorytmy w Pythonie	Jak w języku Python: implementować algorytmy stosując technikę "dziel i zwyciężaj"; korzystać z memoizacji; implementować algorytmy stosując technikę powrotów.
Przechowywanie danych	Jak: czytać i zapisywać dane z/do plików dyskowych; serializować obiekty rożnych typów do postaci tekstowej; używać prostych, słownikowych baz danych.
Korzystanie z modułów PYPI	Jak: korzystać z modułów publikowanych na witrynie Python Package Index; rysować różnego typu wykresy korzystając z modułu matplotlib; generować dokumenty w formatach HTML i PDF; korzystać z relacyjnych baz danych.
Python w zastosowaniach praktycznych	Najważniejsze różnice pomiędzy kodem w języku Python w wersji 2 i 3; jak tworzyć programy z graficznym interfejsem użytkownika korzystając z modułu tkinter; jak przechodzić i sprawdzać zawartość katalogów dyskowych.

4. Rozwiązania metodyczne przyjęte w kursie

Kurs został pomyślany jako prowadzony bez udziału prowadzącego. Motywacja ku temu była dwojaka: z jednej strony, korzystający z niego uczniowie i studenci korzystają z opieki swoich własnych prowadzących, z drugiej zaś, zapewnienie udziału prowadzącego dla wszystkich uczestników kursu przy ograniczonych zasobach kadrowych wymagałoby ograniczenia liczby uczestników kursu, co przeczyłoby oryginalnej intencji udostepnienia kursu wszystkim zainteresowanym.

Za podstawową formę przekazywania wiedzy przyjęto postać tekstową, niekiedy z załączonymi ilustracjami. Ze względu na specyfikę nauki języka programowania, od uczestnika kursu oczekuje się, że w trakcie nauki będzie on samodzielnie przepisywał przykłady kodu rozmieszczone w treści kursu do wskazanego środowiska programistycznego i niezwłocznie je wykonywał, obserwując tego rezultaty.

Aby odbiorca łatwo wychwytywał rozmieszczone w tekście podstawowym przykłady, zastosowano kilka wyraźnie odmiennych formatów tekstu, wyróżniając:

- słowa kluczowe,
- wzorce składniowe,
- poprawne przykłady użycia instrukcji,
- niepoprawne przykłady użycia instrukcji,
- dłuższe fragmenty kodu stanowiące programy lub ich części,
- fragmenty kodu przeznaczone do wykonania w trybie interaktywnym.

Mając na uwadze zorientowanie współczesnej młodzieży na przekaz audiowizualny, 102 jednostki kursu wybrane jako kluczowe dla osiągnięcia celów nauczania, zostały wzbogacone filmami instruktażowymi, zwykle kilkuminutowymi. Ścieżka audio (zarejestrowany głos lektora) każdego z tych filmów odpowiada merytorycznie tekstowi podstawowemu danej jednostki i ewentualnie jednostek sąsiednich, dla których nie zarejestrowano osobnych filmów, natomiast ścieżkę video stanowi zsynchronizowany z tekstem wypowiadanym przez lektora zapis z ekranu przedstawiający pisanie i wykonywanie przykładowego kodu. Dodatkowo w pięciu modułach uznanych za szczególnie ważne umieszczono lekcje video: kilkudziesięciominutowe filmy szkoleniowe o scenariuszu przygotowanym przez innego autora i ujmującym zagadnienia dotyczące danego modułu w innym aspekcie. W założeniu ma pomóc to zrozumieć treści, których podstawowy sposób prezentacji nie przemówił do odbiorcy, a jednocześnie pozwolić na swego rodzaju alternatywne powtórzenie materiału. Z myślą o osobach niesłyszących i niedosłyszących przygotowano napisy do każdego z filmów na platformie.

Elementem mającym służyć współpracy w nauce między uczestnikami kursu jest forum dyskusyjne. Ponieważ pisanie programu wiąże się z rozwiązywaniem problemów dotyczących różnych modułów kursu, zrezygnowano z tworzenia osobnych forów dyskusyjnych dla każdego modułu, a udostępniono jedno wspólne forum dla całego kursu.

Celem bieżącej weryfikacji nabywanej wiedzy, w każdej lekcji kursu umieszczono przynajmniej jedno ćwiczenie. Ćwiczenia w większości są oceniane automatycznie przez platformę (kilka ćwiczeń, polegających na napisaniu krótkiego programu lub tekstu, jest ocenianych przez innych uczestników kursu na zasadzie peer assessment). Większość ćwiczeń ocenianych automatycznie ma formę puzzli rozwiązywanych na zasadzie przeciągania i upuszczania fragmentów kodu, tak by powstał z nich poprawny program, rzadziej użyto pytań otwartych (zwykle dotyczą podanie nazwy właściwej instrukcji), sporadycznie zdarzają się pytania zamknięte jednokrotnego wyboru.

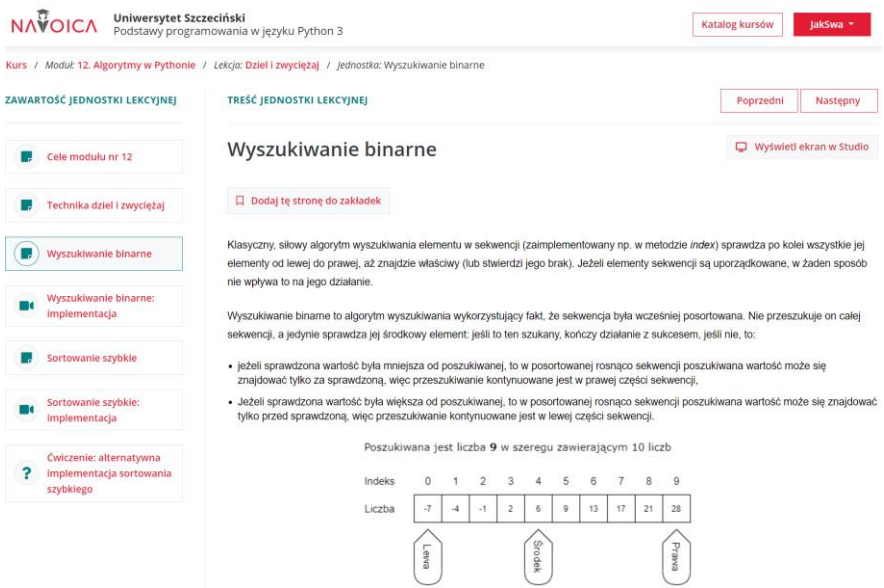
Wyniki oceny ćwiczeń nie są zaliczane do oceny końcowej z kursu, o której decydują wyłącznie wyniki uzyskane w testach podsumowujących moduły i teście zamknięcia. Ustalono, że każdy uczestnik kursu, który udzieli poprawnych odpowiedzi na co najmniej 75% pytań zawartych w tych testach, otrzyma elektroniczne zaświadczenie o ukończeniu kursu z wynikiem pozytywnym. Nie postawiono osob-

nych wymogów dla testów podsumowujących moduły i zamknięcia, wychodząc z założenia, że uczestnicy przykładający się do systematycznej nauki (uzyskujący zatem dobre wyniki w testach podsumowujących moduły) mogą potraktować ulgowo test zamknięcia, z kolei ci, którym w trakcie kursu nie szło najlepiej, mają szansę przygotować się lepiej do testu zamknięcia, by zdobyć brakujące punkty.

Należy zauważyć, iż przyjęte rozwiązania metodyczne są w pełni zgodne z wynikami badań B.T. Wonga [21] dotyczących czynników, które w kursach MOOC wpływają najmocniej odpowiednio na zachętę do nauki (szczegółowe wprowadzenie – wskazało je 100% respondentów), zaangażowanie (dostępność multimediów – 97%), interakcję online (forum dyskusyjne – 100%) i konsolidację wiedzy (automatycznie oceniane testy – 81%); elementy pominięte w kursie były wskazywane przez znacznie mniejszy odsetek respondentów, np. prowadzenie transmisji na żywo (w kontekście zaangażowania) wskazało tylko 3% respondentów [21].

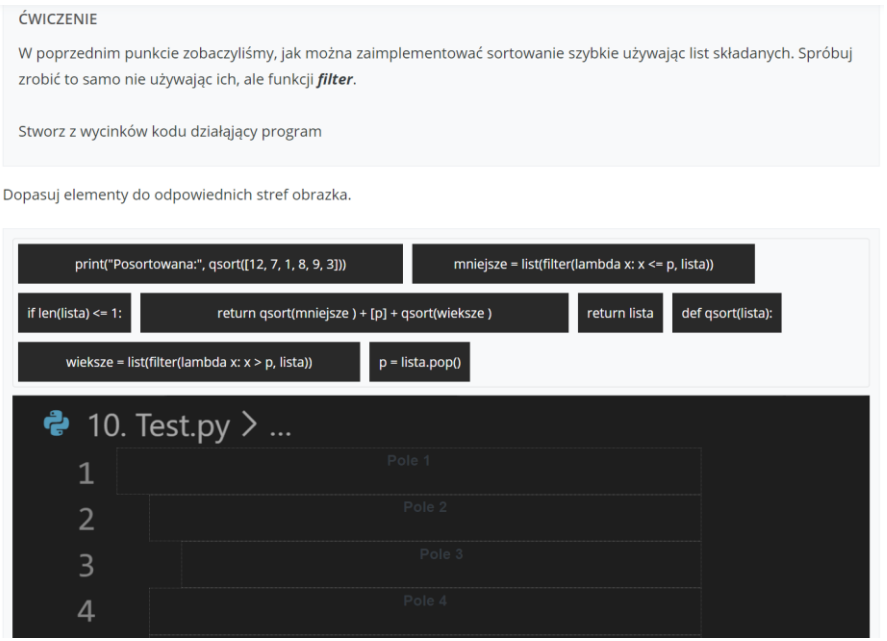
5. Implementacja kursu

Zgodnie z wymogami konkursu „Kurs na MOOC" [7], implementacji kursu dokonano na platformie navoica.pl [11]. Platforma ta oparta jest na oprogramowaniu Open edX [4] i od strony technicznej umożliwia publikowanie wszystkich rodzajów treści i ćwiczeń przewidzianych w opisywanym kursie. Dla przybliżenia wyglądu kursu, na rys. 1 zaprezentowano przykładową jednostkę kursu, a na rys. 2 przykładowe ćwiczenie typu „przeciągnij i upuść".



Rysunek 1. Przykładowa jednostka kursu (treść podstawowa)

Przygotowanie i implementacja kursu zajęło łącznie ponad 900 roboczogodzin, co oznacza 15 godzin pracy przypadające na 1 godzinę kursu, a więc ok. 36% średniej (42 h) ustalonej w wyniku badań przeprowadzonych w 2017 r. przez K. Kappa i R. Defelice [1]. Z pewnością ponadprzeciętnie szybki czas opracowania kursu możliwy był dzięki wykorzystaniu jako wyjściowej bazy materiałów przygotowanych na potrzeby wcześniejszych kursów programowania w języku Python niniejszego autora [16,17].



Rysunek 2. Przykład ćwiczenia typu „przeciągnij i upuść”

6. Wstępna ewaluacja kursu

Jakkolwiek właściwa ewaluacja kursu będzie mogła nastąpić dopiero po zakończeniu jego pierwszej edycji, dążąc do zapewnienia wysokiej jakości kursu przed jego opublikowaniem przeprowadzono ewaluację wstępną. Podzielono ją na dwie równolegle realizowane (czerwiec 2020) ścieżki. Do udziału w pierwszej z nich zaproszono łącznie 30 studentów kierunków Informatyka i ekonometria oraz Informatyka w biznesie prowadzonych na Wydziale Ekonomii, Finansów i Zarządzania Uniwersytetu Szczecińskiego. Mieli oni korzystać z kursu jako beta testerzy i raportować wszystkie zidentyfikowane przez siebie jego wady merytoryczne lub techniczne. Mimo iż dla większego zaangażowania studentów każdą nowo zgłoszoną

wadę nagradzano punktami wpływającymi na podwyższenie oceny semestralnej, uczestniczący w ewaluacji studenci zdołali wykryć jedynie cztery wady.

Do udziału w drugiej z nich zaproszono dwoje ekspertów z dwóch polskich uczelni (prof. dr hab. i dr inż., oboje mający wieloletnie doświadczenie w prowadzeniu kursów e-learningowych), którzy także mieli korzystać z kursu jako beta testerzy, a następnie ocenić kurs przy pomocy formularza ewaluacyjnego. Formularz składał się z dwóch części: pierwszej dotyczącej aspektów formalno-technicznych, liczącej 44 pytania zamknięte i 6 otwartych, oraz drugiej, obejmującej aspekty metodyczne i merytoryczne kursu, w której zawarto 14 pytań zamkniętych i 5 otwartych. Oboje eksperci w pełni pozytywnie ocenili kurs, tak od strony technicznej, jak i metodycznej i merytorycznej, nie wymagając wprowadzenia żadnych poprawek.

7. Podsumowanie

Opisany wyżej kurs, dostępny bezpłatnie dla wszystkich zainteresowanych [11], wychodzi naprzeciw rosnącemu zainteresowaniu nauką programowania w języku Python i stanowi próbę zapełnienia luki istniejącej w polskim Internecie. Pomyślany z myślą głównie o studentach studiów wyższych, został jednak opracowany w sposób pozwalający na jego wykorzystanie także na innych szczeblach edukacji (w szczególności w szkołach średnich), a także przez osoby samouczące się.

Kurs obejmuje 14 obszarów tematycznych, prowadząc od podstaw korzystania z języka Python i jego środowiska interaktywnego, poprzez prezentację składni i zestawu instrukcji tego języka, na jego zastosowaniu do rozwiązywania problemów (tak podręcznikowych, jak i praktycznych) kończąc.

Kurs zawiera automatycznie oceniane ćwiczenia i pytania testowe, a jego pozytywne ukończenie nagradzane jest stosownym zaświadczeniem.

Rozpoczynająca się właśnie pierwsza edycja kursu pozwoli zidentyfikować jego ewentualne niedoskonałości, których nie dostrzeżono w ramach wstępnej ewaluacji, oraz kierunki możliwych udoskonaleń.

Literatura

1. Defelice, R., How Long to Develop One Hour of Training? Updated for 2017, https://www.td.org/insights/how-long-does-it-take-to-develop-one-hour-of-training-updated-for-2017, ostatni dostęp 4.09.2020 roku.
2. Interaktywny samouczek w języku Python 3, https://snakify.org/pl/, ostatni dostęp 4.09.2020 roku.
3. Introduction to Computer Science and Programming Using Python, https://www.edx.org/course/introduction-to-computer-science-and-programming-7, ostatni dostęp 4.09.2020 roku.

4. Kaczmarek-Kacprzak, A., Kurowska-Wilczyńska, K., *Navoica – Polski MOOC*, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej, 65, s. 35-42, 2019.
5. Kaczmarek-Kacprzak, A., Lewicki, J., Muczyński, B., Szreniawa–Sztajnert, A., *Moc MOOC-ów – Czas na polskie rozwiązania systemowe*, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej, 41, s. 21-26, 2015.
6. Klobas, J.E., Mackintosh, B., Murphy, J., The Anatomy of MOOCs, w: *Massive Open Online Courses. The MOOC Revolution*, Kim P. (ed.), s. 1-22, Routledge, New York, 2015.
7. Konkurs „Kurs na MOOC", https://navoica.edu.pl/konkurs-kurs-na-mooc, ostatni dostęp 4.09.2020 roku.
8. Mobilo, R., Python dla początkujących, https://www.udemy.com/course/python-dla-poczatkujacych, ostatni dostęp 4.09.2020 roku.
9. Pełszyński, Ł, Python. Podstawy programowania, https://kodologia.pl/kursy/python-podstawy-programowania, ostatni dostęp 4.09.2020 roku.
10. Pierwsze kroki w języku Python, https://riptutorial.com/pl/python, ostatni dostęp 4.09.2020 roku.
11. Podstawy programowania w języku Python 3, https://navoica.pl/courses/course-v1:US+PP1+2020_1/course/, ostatni dostęp 4.09.2020 roku.
12. Python Essentials, https://www.netacad.com/portal/web/self-enroll/c/course-976146, ostatni dostęp 4.09.2020 roku.
13. Severance C.R., Programming for Everybody (Getting Started with Python), https://www.coursera.org/learn/python, ostatni dostęp 4.09.2020 roku.
14. Shah, D., MOOCs Started Out Completely Free. Where Are They Now?, https://www.classcentral.com/report/moocs-started-completely-free-now/, ostatni dostęp 4.09.2020 roku.
15. Shen, R., Lee, M.J., Learners' Perspectives on Learning Programming from Interactive Computer Tutors in a MOOC, w: *2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, s. 1-5, IEEE, Dunedin, 2020.
16. Swacha, J., Development and evaluation of an interactive Python course, w: *ICERI2018 Proceedings*, s. 456-466. IATED, Sewilla, 2018.

17. Swacha, J., Python, http://uoo.univ.szczecin.pl/~jakubs/, ostatni dostęp 4.09.2020 roku.
18. TIOBE Index for August 2020, https://www.tiobe.com/tiobe-index/, ostatni dostęp 4.09.2020 roku.
19. Web Programming with Python and JavaScript, https://www.edx.org/course/cs50s-web-programming-with-python-and-javascript, ostatni dostęp 4.09.2020 roku.
20. Włodarczyk, A., Python 3 od Podstaw do Eksperta, https://www.udemy.com/course/python-od-podstaw-dla-poczatkujacych, ostatni dostęp 4.09.2020 roku.
21. Wong, B.T., *Factors leading to effective teaching of MOOCs*, Asian Association of Open Universities Journal 11(1), 2016.