motywy Logo

Nowe obszary programowania

Andrzej WALAT

Pierwsze komputery, w latach czterdziestych XX wieku, były używane głównie do złożonych obliczeń numerycznych. Również pierwsze wysokopoziomowe języki programowania, jak na przykład bardzo popularny przez wiele lat FORTRAN, były tworzone głównie na potrzeby obliczeń numerycznych.

Z czasem zakres zastosowań komputerów i programowania zaczął się stopniowo poszerzać. Pojawiły się i zaczęły nabierać znaczenia nowe rodzaje zadań programistycznych, a w związku z tym powstały nowe metody i języki programowania. Zmiany były tak zasadnicze, że dla podkreślenia ich wagi mówimy czasem o nowych **paradygmatach programowania**. Rozwój badań nad sztuczną inteligencją był impulsem do powstania metod i języków **programowania funkcyjnego**, takich jak LISP i od niego pochodne, np. Scheme, Logo i wiele innych. Wzrost znaczenia zadań sterowania doprowadził do powstania **programowania zorientowanego obiektowo** i odpowiednich języków programowania obiektowego, takich jak SIMULA i SMALLTALK, a w końcu do tego, że dziś prawie każdy zaawansowany język programowania jest obiektowy. Ale na tym nie koniec. Pojawiają się i nadal będą się pojawiały coraz to nowe obszary zastosowań programowania. W tym artykule ograniczę się do dwóch obszarów, jakimi są **uczenie się** oraz **twórczość plastyczna**.

Uczenie się i programowanie

Logo ma w Polsce opinię elementarnego języka do nauki podstaw programowania. Mało kto wie, że Papert z zespołem współpracowników z MIT stworzyli Logo głównie jako **środowisko uczenia się** bardzo różnorodnych umiejętności, nie tylko i nie na pierwszym miejscu – umiejętności programowania. Jednym z moich celów jako autora kolumny *Logomotywy* jest ukazanie roli Logo jako środowiska uczenia się. Wiele interesujących przykładów i głębokich uwag na temat uczenia się przez programowanie można znaleźć w książkach [1] i [2].

Stephen Wolfram we wstępie do głośnej książki *A New Kind of Science* napisał: „Trzy wieki temu nastąpił przełom w nauce dzięki rewolucyjnemu odkryciu, że reguły w postaci równań matematycznych mogą być użyteczne do opisu świata przyrodniczego. Celem tej książki jest zapoczątkowanie równie wielkiego przełomu, zapoczątkowanie nowej nauki opartej na znacznie bardziej ogólnych regułach, które można wyrażać w formie prostych programów komputerowych."

Choć może nie wszyscy podzielają zdanie, że dzieło Wolframa jest tak przełomowe, jak uważa autor, to jednak jego główna myśl, iż programy komputerowe mogą być równie dobrymi (a często są znacznie lepszymi) modelami rzeczywistości, jak tradycyjne już równania i nierówności, jest dzisiaj akceptowana dość powszechnie. Skoro tak, to każdy powinien uczyć się w jakimś zakresie programowania, tak jak dziś każdy uczy się równań i nierówności. Posługiwanie się równaniami jest nie tylko celem, lecz również warunkiem dalszej skutecznej nauki. To samo można powiedzieć o umiejętności programowania.

Twórczość plastyczna i programowanie

Powszechnie wiadomo, że wspominany wyżej MIT – Massachusetts Institute of Technology – jest jedną z najbardziej renomowanych na świecie uczelni (w zakresie kształcenia najwyższej klasy profesjonalnych programistów prawie tak dobry jak Uniwersytet Warszawski). Ale nie wszyscy wiedzą, że chlubą MIT jest Media Laboratory – wydział projektowania artystycznego kształcący nowoczesnych designerów (zobacz [5]). John Maeda, guru tego wydziału, jednej ze swoich książek nadał tytuł *Creative Code – Twórczy program*. Oczywiście, tytuł książki jest skrótem myślowym. John Maeda dobrze wie, że prawdziwym twórcą komputerowego dzieła plastycznego jest autor programu, a nie program lub komputer. Współcześni artyści i designerzy posługują się nie tylko komputerowymi wirtualnymi ołówkami, pędzlami i rozpylaczami, ale również piszą programy, które tworzą dzieła artystyczne. Ale na razie nie powstały jeszcze (o ile wiem) publikacje próbujące określić zasady dobrego programowania w tej dziedzinie. Jakość programu ocenia się wyłącznie według walorów artystycznych dzieła.

Programowanie i alfabetyzacja

Istnieje wiele analogii między współczesnym postępem znaczenia programowania i historią rozwoju czytelnictwa i piśmiennictwa. Czytanie i pisanie było aż do późnego średniowiecza zajęciem bardzo nielicznej grupy profesjonalistów: mnichów kopiujących święte księgi, kronikarzy i kancelistów królewskich, krótko mówiąc – skrybów. Odkrycie druku i metod produkcji taniego papieru spowodowało, że bardzo szybko stało się umiejętnością potrzebną każdemu. Wraz z wkraczaniem piśmiennictwa na nowe obszary wykrystalizowało się wiele istotnie różnych zakresów języka pisanego: język literacki, prasowy, naukowy, prawniczy itd. Można świetnie funkcjonować w jednym zakresie i być kompletnym analfabetą w innych.

Stworzenie metod programowania funkcyjnego oraz programowania zorientowanego obiektowo było istotnym postępem w rozwoju informatyki, ale programowanie nadal było zajęciem profesjonalnych programistów (skrybów). Pojawienie się nowych obszarów zastosowań programowania, takich jak, na przykład, uczenie się i twórczość plastyczna, powodują, że staje się ono stopniowo zajęciem coraz większej liczby grup społecznych. Podobnie jak istnieją różne zakresy języka pisanego i różne związane z nimi subkultury językowe, istnieją też różne kultury programowania.

Literatura

[1] Abelson H., Di Sessa A. (1992), *Geometria żółwia*, WNT, Warszawa.
[2] Di Sessa A. (2000), *Changing Minds, Computers, Learning, and Literacy*, A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England.
[3] Maeda J. (2004), *Creative Code*, Thames & Hudson, London.
[4] Wolfram S. (2002), *A New Kind of Science*, Wolfram Media, Inc.
[5] `www.media.mit.edu`