Informatyka w Edukacji, XV
UMK Toruń, 2018

„UMIEM PROGRAMOWAĆ", WIĘC UMIEM ROZWIĄZYWAĆ PROBLEMY. O PROJEKCIE UNIJNYM REALIZOWANYM WŚRÓD ŁÓDZKICH UCZNIÓW KLAS VII – VIII

Karolina Ryżowicz
Learnetic SA
karolina.ryzowicz@learnetic.com http://www.learnetic.pl

Abstract. Coding is a difficult task for students and teachers as well. In order to increase students' willingness to create programs, it is necessary to make the learning process an exciting game for them. With this in mind, we have created a coding course based on an interesting plot which gradually introduces students to the programming world. The educational programming language created specially for this course enables an easy transition from visual programming to text one and gives young programming enthusiasts a good start to learning a particular programming language.

1. Wstęp

"Umiem programować" to projekt realizowany w ramach Regionalnego Programu Operacyjnego Województwa Łódzkiego 2014-2020. Jego liderem jest Łódzkie Centrum Doskonalenia Nauczycieli i Kształcenia Praktycznego. Wsparciem zostały objęte wybrane gimnazja z Łodzi, nauczyciele i uczniowie tych szkół. W ramach projektu doposażono szkoły w sprzęt i aplikacje TIK oraz stworzono cykl dziesięciu interaktywnych aplikacji edukacyjnych przygotowujących uczniów klas VII i VIII do nauki programowania .

Głównym celem kursu „Umiem programować" jest kształtowanie zdolności algorytmicznego myślenia, będące pierwszym krokiem do zdobycia umiejętności programowania. Uczniowie korzystając z kierowanego do nich kursu, kształtują wymienione umiejętności zgodnie z założeniami nowej podstawy programowej kształcenia informatycznego. W algorytmicznym rozwiązywaniu problemu dokonują jego specyfikacji, zapisują rozwiązanie w postaci programu, objaśniają przebieg jego działania oraz testują je dla przykładowych danych.

Założeniem kursu jest nie tylko pisanie programów, ale przede wszystkim budowanie kompetencji cyfrowych, rozwijanie logicznego myślenia oraz uczenie kreatywnego podejścia do rozwiązywania sytuacji problemowych z różnych dziedzin.

2. Elementy grywalizacji

Kurs jest oparty na bogatej fabule, której motyw przewodni stanowi powrót z wakacji na słonecznej wyspie Skryptos. W wyniku awarii statek Morska Pętla zbacza z kursu i dopływa do brzegów nieznanej wyspy, gdzie czekają na nas niezapomniane przygody. Wędrujemy szlakiem zakorkowanych butelek, pomagamy Argumentanom w ich codziennej pracy, odwiedzamy Instytut Historii Jakiej Nie Znacie, zostajemy uwięzieni w kamiennym kręgu, rozwiązujemy zagadki Tablicesa, żeby wydostać się z mrocznego lasu, docieramy do zaginionego miasta Kompiloseum, gdzie bierzemy udział w Festiwalu Algorytmiki, a nawet organizujemy akcję ratunkową w górach.

Rysunek 1 Mapa przygody przedstawiająca kolejne lokacje

Każda lekcja zawiera film wprowadzający w sytuację fabularną. Informuje on, w jakiej przygodzie bierzemy udział oraz jaki cel musimy osiągnąć.

Rysunek 2 Przykładowy film przedstawiający sytuację fabularną

Cel główny realizujemy metodą małych kroków. Za każde poprawnie rozwiązane zadanie programistyczne otrzymujemy nagrodę (pamiątkową monetę zmiennar, karteczkę z zagadkową wskazówką, pieczątkę będącą częścią sudoku, itd.) lub odblokowujemy potrzebny przedmiot.

Rysunek 3 Przykładowy pasek nawigacji pokazujący stan lekcji

Zadania programistyczne są przeplatane dodatkowymi minizadaniami aktywizującymi ucznia. Mają one postać karty zadań, mapy lub scenki komiksowej i pełnią rolę nawigacji po lekcji.

Rysunek 4 Przykładowy pasek nawigacji pokazujący stan lekcji

W każdej lekcji po rozwiązaniu ostatniego zadania w formie gry (memo, sudoku, wykreślanka, "wisielec", puzzle, itd.) możemy obejrzeć animację przedstawiającą osiągnięcie celu, jaki nam przyświecał. Dodatkowym elementem spajajacym wszystkie lekcje jest postać profesora Kodera, którego spotykamy podczas pierwszej przygody. Profesor staje się naszym mentorem i towarzyszy nam w dalszej podróży. W trakcie video-rozmów wprowadza nas w coraz bardziej zaawansowany świat programowania, niejednokrotnie pomagając nam wyjść cało z opresji.

Rysunek 5 Widok video-rozmowy z profesorem Koderem

Wcielanie się w głównego bohatera, przeżywanie jego przygód, przy jednoczesnym podejmowaniu działań prowadzących do osiągnięcia wyznaczonych celów wzbogaca proces kształcenia o elementy grywalizacji i sprawia, że staje się on pasjonującą zabawą.

3. Edukacyjny język programowania

Na potrzeby projektu został stworzony edukacyjny język programowania, którego struktura jest zbliżona do Pascala. Wszystkie słowa kluczowe utworzono w języku polskim, co sprawia, że proces nauki jest łatwiejszy i bardziej komfortowy dla ucznia. Kolejne ułatwienie stanowi rezygnacja ze ścisłych reguł deklaracji zmiennych. W efekcie sprowadza się ona do podania wyłącznie nazwy zmiennej. Do obustronnej komunikacji z wykonywanym programem służą instrukcje pisz() oraz czytaj().

```
1 program powitanie
2 zmienna imie
3 poczatek
4 pisz("Jak masz na imię? ")
5 imie = czytaj()
6 pisz("Cześć " +imie +"!")
7 nowa_linia()
8 pisz(":)")
9 koniec
```

```
Jak masz na imię? Koder
Cześć Koder!
:)
```

Rysunek 6 Struktura programu

Język jest wyposażony we wszystkie operatory arytmetyczne, relacyjne i logiczne oraz instrukcje warunkowe i iteracyjne, dzięki czemu pozwala na precyzyjny zapis algorytmów realizujących obliczenia. Ponadto, instrukcja odpowiedz(), użyta wewnątrz programu, pozwala na sprawdzenie jego poprawności. Jej działanie sprowadza się do porównania wyników otrzymanych przez program po wykonaniu obliczeń z wartościami zadeklarowanymi jako poprawne odpowiedzi.

```
1  program zakupy
2  zmienna kwota, rabat, oszczedzasz, do_zaplaty
3  poczatek
4  pisz("Podaj wartość swoich zakupów: ")
5  kwota=czytaj()
6  jesli kwota<100 to
7  rabat=0.1
8  przeciwnie
9  jesli kwota>=100 oraz kwota<200 to
10 rabat=0.15
11 przeciwnie
12 rabat=0.2
13 oszczedzasz = kwota * rabat
14 do_zaplaty = kwota - oszczedzasz
15 pisz("Przysługuje ci rabat " + rabat*100 + "%.")
16 nowa_linia()
17 pisz("Zaoszczędzasz " + oszczedzasz + " zł.")
18 nowa_linia()
19 pisz("Do zapłaty zostało " + do_zaplaty + " zł.")
20 odpowiedz(oszczedzasz,do_zaplaty)
21 koniec
```

```
Podaj wartość swoich zakupów: 240
Przysługuje ci rabat 20%.
Zaoszczędzasz 48 zł.
Do zapłaty zostało 192 zł.
```

Rysunek 7 Przykładowy program realizujący obliczenia matematyczne

Dodatkowo, język oferuje możliwość skorzystania z funkcji losuj(), która zwraca losowo wybraną liczbę całkowitą z podanego przedziału, funkcji reszta(), która zwraca resztę z dzielenia podanych liczb oraz funkcji ile_razy(), która zaokrągla wynik dzielenia do całości (w dół).

```
1  program dzialania
2  zmienna liczba1, liczba2, liczba3
3  poczatek
4  liczba1 = losuj(1,50)
5  liczba2 = ile_razy(liczba1,4)
6  liczba3 = reszta(liczba1,4)
7  pisz("liczba1 to " +liczba1)
8  nowa_linia()
9  pisz("liczba2 to " +liczba2)
10 nowa_linia()
11 pisz("liczba3 to " +liczba3)
12 nowa_linia()
13 pisz(liczba1 +" : 4 = " +liczba2 +" r " +liczba3)
14 koniec
```

```
liczba1 to 34
liczba2 to 8
liczba3 to 2
34 : 4 = 8 r 2
```

Rysunek 8 Program prezentujący działanie dodatkowych funkcji

Ważnym i jednocześnie trudnym dla ucznia elementem nauki programowania jest budowanie podprogramów. Omawiany język daje możliwość zadeklarowania własnych funkcji z parametrami oraz bez parametrów. Deklaracja funkcji ma miejsce na zewnątrz programu.

```
1  funkcja zegar(godziny,minuty,sekundy)
2  zmienna wynik
3  poczatek
4  wynik=godziny*60*60
5  wynik=wynik+minuty*60
6  wynik=wynik+sekundy
7  zwroc(wynik)
8  koniec
9
10 program zamiana_na_sekundy
11 zmienna godziny,minuty,sekundy
12 poczatek
13 pisz("godziny: ")
14 godziny=czytaj()
15 pisz("minuty: ")
16 minuty=czytaj()
17 pisz("sekundy: ")
18 sekundy=czytaj()
19 pisz("Czas wyrażony w sekundach: ")
20 zegar(godziny,minuty,sekundy)
21 koniec
```

```
godziny: 3
minuty: 14
sekundy: 27
Czas wyrażony w sekundach: 11667
```

Rysunek 9 Przykład funkcji z parametrem

Język umożliwia również deklarację tablic statycznych. W celu ułatwienia i skrócenia zapisu programów, lista dostępnych instrukcji została rozbudowana o niestandardowe polecenia pozwalające w szybki i wygodny sposób wypełnić tablice losowo wybranymi lub kolejnymi liczbami z podanego przedziału.

Uczeń stopniowo poznaje kolejne konstrukcje i możliwości języka uczeń. Każda lekcja (poza pierwszą) zawiera film objaśniający zagadnienia teoretyczne, których znajomość jest potrzebna do rozwiązania zadań. Ma on formę screencastu prezentującego poznawane instrukcje języka.

```
program tablice
tablica pierwsza[10]
tablica druga[10]=[1,2,3,4]
tablica trzecia[10]
tablica czwarta[10]
poczatek
pisz(pierwsza)
nowa_linia()
pisz(druga)
nowa_linia()
trzecia.wypelnij_losowo(1,100)
pisz(trzecia)
nowa_linia()
czwarta.wypelnij_kolejno(17)
pisz(czwarta)
koniec
```

```
0 0 0 0 0 0 0 0 0 0
1 2 3 4 0 0 0 0 0 0
59 77 11 84 76 97 6 29 55 54
17 18 19 20 21 22 23 24 25 26
```

Rysunek 10 Wypełnianie tablic za pomocą gotowych poleceń

```
program petle
zmienna licznik1, licznik2
poczatek
dla licznik1 od 1 do 3 co 1 wykonuj
 poczatek
 dla licznik2 od 1 do 5 co 1 wykonuj
  poczatek
  pisz(licznik2 + " ")
  odpowiedz(licznik2)
  koniec
 nowa_linia()
 koniec
```

uruchom

```
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
```

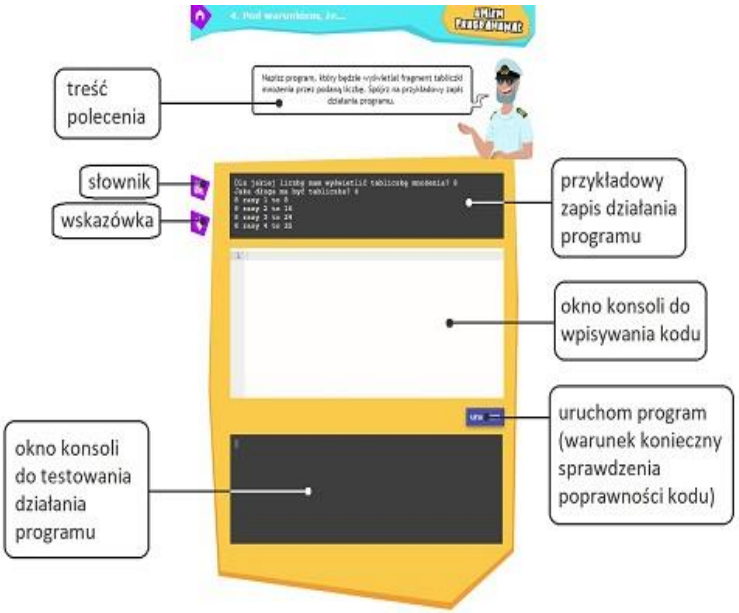
Rysunek 11 Screencast prezentujący poznawane instrukcje języka

4. Zakres merytoryczny

Materiał kursu podzielono na logicznie uporządkowane bloki tematyczne, w obrębie których zadania dobrano według zasad stopniowania trudności. Kolejne lekcje kształtują i utrwalają umiejętności stosowania:

- podstawowych pojęć z zakresu algorytmiki,
- algorytmów liniowych w celu sterowania ruchem postaci (lub pisaka) po planszy,
- algorytmów warunkowych z wykorzystaniem instrukcji wejścia/wyjścia oraz operatorów arytmetycznych, relacyjnych i logicznych,
- algorytmów iteracyjnych,
- zagnieżdżonych instrukcji warunkowych i iteracyjnych
- funkcji z parametrami i bez parametrów,

- funkcji rekurencyjnych,
- zmiennych i tablic,
- podstawowych algorytmów przeszukiwania i sortowania zbiorów danych.



Rysunek 12 Przykładowy układ strony z zadaniem programistycznym

Po uruchomieniu programu uczeń otrzymuje komunikat o ewentualnych błędach oraz ich typie (np. „Błąd składni językowej"). Lekcje rozwijają zatem również umiejętności diagnozowania i poprawiania popełnionych błędów.

```
Wykorzystanie niezdefiniowanej zmiennej 'liczba'
```

Rysunek 13 Przykładowy komunikat o błędzie

W trakcie rozwiązywania zadań uczeń może korzystać ze słownika zawierającego wszystkie poznane instrukcje języka oraz z licznych podpowiedzi i wskazówek. Ma także możliwość zobaczenia poprawnej odpowiedzi, jednak wyświetlony gotowy program nie może być skopiowany.

W doborze zadań uwzględniono:

- algorytmy wymienione w nowej podstawie programowej kształcenia informatycznego, m.in. algorytm Euklidesa w obu wersjach iteracyjnych, obliczanie

średniej arytmetycznej, obliczanie reszty z dzielenia, badanie podzielności liczb,

- zagadnienia interdyscyplinarne, m.in. przeliczanie temperatury wyrażonej w stopniach Fahrenheita na stopnie Celsjusza, zamianę liczb zapisanych w systemie dziesiętnym na binarne, rozpoznawanie sygnału SOS,
- sytuacje z życia codziennego, m.in. obliczanie czasu podróży i kosztu zużytego paliwa, kosztu zakupów z uwzględnieniem rabatu i otrzymanej reszty, liczby płytek podłogowych potrzebnych do remontu, kaloryczności zestawu obiadowego.

Każda lekcja kończy się quizem podsumowującym i utrwalającym poznane instrukcje oraz umiejętność ich stosowania. Po poprawnym rozwiązaniu quizu uczeń może przeczytać ciekawostkę dotyczącą programowania.

5. Modele pracy

Kurs "Umiem programować" jest przeznaczony zarówno do pracy indywidualnej, jak i grupowej. Forma kursu umożliwia jego wykorzystanie przez nauczycieli w trakcie lekcji do wprowadzenia, utrwalenia i sprawdzenia materiału, a także podczas pracy własnej uczniów w domu. Zagadnienia teoretyczne przedstawione w formie przejrzystego screencastu oraz liczne wskazówki i podpowiedzi dają podstawę do podjęcia całkowicie samodzielnych prób programowania.

Praca z „Umiem programować" pozwala na indywidualizację procesu nauczania poprzez prowadzenie zajęć z podziałem na grupy. Nauczyciel może podzielić klasę na zespoły o zróżnicowanych umiejętnościach, dzięki czemu uczniowie mający mniejsze predyspozycje do programowania będą współpracować z uczniami bardziej uzdolnionymi i w ten sposób poszerzać swoją wiedzę. Tutoring rówieśniczy wpływa na budowanie kompetencji kluczowych oraz pozwala na aktywizację wszystkich uczniów, co sprawia, że nabierają oni wiary we własne możliwości. Drugim rozwiązaniem jest podział klasy na zespoły jednorodne, w których uczniowie prezentują podobny poziom umiejętności. W wyniku zastosowania takiego podejścia nauczyciel może więcej uwagi poświęcić uczniom o mniejszych predyspozycjach programistycznych, pozwalając jednocześnie na większą samodzielność uczniów uzdolnionych. Kurs sprawdza się również w roli materiału do przeprowadzenia zajęć pozalekcyjnych dla uczniów.

6. Podsumowanie

Nauka programowania w szkole to trudne zadanie, zarówno dla uczniów jak i nauczycieli. Aby rozbudzić w uczniach chęć tworzenia programów, należy stwo-

rzyć im takie warunki, aby ten proces był pasjonującą zabawą. Kurs „Umiem programować", dzięki zastosowaniu animacji, komiksów, gier dydaktycznych oraz quizów w połączeniu z nowoczesną i dostosowaną do współczesnych trendów szatą graficzną, wzbudza zainteresowanie uczniów i inspiruje ich do rozwiązywania problemów w sposób programistyczny. Stworzony na potrzeby kursu edukacyjny język programowania umożliwia łagodne przejście od programowania wizualnego do tekstowego i daje dobry start do nauki konkretnego języka programowania.

Programowanie to nie tylko modne hasło, to wymóg obecnych czasów. W każdej dziedzinie życia rośnie zapotrzebowanie na programistów. Kurs „Umiem programować" kształtuje umiejętność algorytmicznego myślenia i daje wielu młodym entuzjastom programowania szansę na to, aby pogłębić swoje zainteresowania w tej dziedzinie.