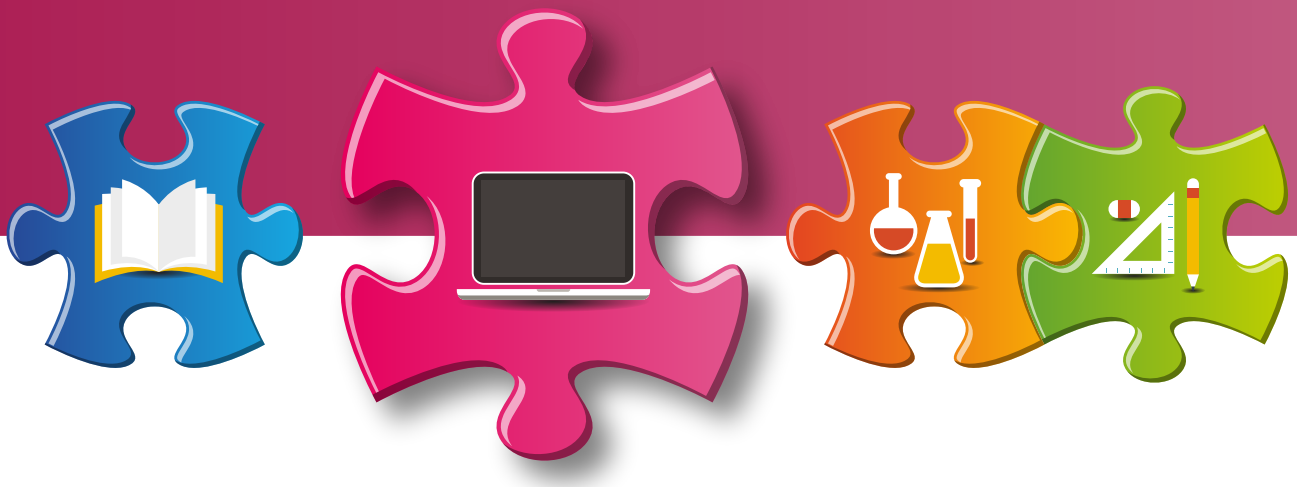
Zestaw 7
Zeszyt 2

Jacek Stańdo
Monika Spławska-Murmyło

Pomysły wykorzystania wizualnych języków programowania w edukacji informatycznej dzieci starszych i młodzieży

- Przegląd narzędzi do programowania wizualnego



Redakcja językowa i korekta
Anna Wawryszuk

Projekt graficzny, projekt okładki
Wojciech Romerowicz, ORE

Skład i redakcja techniczna
Grzegorz Dębiński

Projekt motywu graficznego „Szkoły ćwiczeń”
Aneta Witecka

ISBN 978-83-65890-47-4 (Zestawy materiałów dla nauczycieli szkół ćwiczeń – informatyka)
ISBN 978-83-65890-76-4 (Zestaw 7. Rozwój myślenia komputacyjnego w klasach IV–VIII szkoły podstawowej)
ISBN 978-83-65890-78-8 (Zeszyt 2. Pomysły wykorzystania wizualnych języków programowania w edukacji informatycznej dzieci starszych i młodzieży)

Warszawa 2017
Ośrodek Rozwoju Edukacji
Aleje Ujazdowskie 28
00-478 Warszawa
www.ore.edu.pl

Fundusze Europejskie Wiedza Edukacja Rozwój
ORE Ośrodek Rozwoju Edukacji
Unia Europejska Europejski Fundusz Społeczny

Spis treści

Wstęp

Według nowej podstawy programowej w klasach IV–VIII szkoły podstawowej uczniowie odbywają aktywny kurs wprowadzającego i rozwijającego umiejętności programowania. Idealną sytuacją jest, kiedy uczniowie znają już podstawowe pojęcia związane z informatyką. Należą do nich np. liniowa kolejność (sekwencja zdarzeń, logiczny porządek zdarzeń, czynności i wielkości), instrukcja (polecenie), algorytm (plan działania), a także mieli do czynienia z wizualizacją lub symulacją działań algorytmicznych (w formie zabawy). Najczęściej jednak spotykamy się z sytuacją inicjalną: w klasie IV szkoły podstawowej uczniowie dopiero zaczynają naukę programowania.

Kurs w klasach IV–VIII obejmuje pracę w co najmniej jednym języku programowania, najczęściej jest to Scratch, Pascal, Baltie lub Python. Kończąc szkołę podstawową, uczeń powinien:

1. projektować, tworzyć i zapisywać w wizualnym języku programowania:
    - pomysły historyjek i rozwiązania problemów, w tym proste algorytmy z wykorzystaniem poleceń sekwencyjnych, warunkowych i iteracyjnych oraz zdarzeń jednoczesnych;
    - prosty program sterujący robotem lub innym obiektem na ekranie komputera;
2. testować na komputerze swoje programy pod względem zgodności z przyjętymi założeniami i ewentualnie je poprawiać, objaśniać przebieg działania programów (Podstawa programowa…, b.r.).

Podkreślmy, że programowanie jest tu rozumiane jako pisanie programu w określonym języku programowania. Przez odniesienie do wiedzy i umiejętności z zakresu myślenia komputacyjnego programowanie zyskuje dodatkowy wymiar: jest rozumiane jako proces składający się z działań zmierzających do rozwiązania problemu. Elementami tego procesu są:

- specyfikacja problemu,
- znalezienie i opracowanie rozwiązania,
- zaprogramowanie rozwiązania,
- przetestowanie jego poprawności,
- dokonanie ewentualnej korekty.

Uczeń po szkole podstawowej powinien już programować algorytmy na liczbach naturalnych: badać podzielność liczb, wyodrębniać cyfry danej liczby, przedstawiać działanie algorytmu Euklidesa w obu wersjach iteracyjnych (z odejmowaniem i z resztą z dzielenia). Od ucznia można oczekiwać również, że stosuje algorytmy wyszukiwania i porządkowania: wyszukuje element w zbiorze uporządkowanym i nieuporządkowanym oraz porządkuje elementy w zbiorze metodą prostego wybierania i zliczania. W zakresie programowania w robotyce uczeń powinien projektować, tworzyć i testować oprogramowanie sterujące robotem lub innym obiektem na ekranie lub w rzeczywistości.

Warunki i sposób realizacji (wypis z komentarza do podstawy programowej)

Od klasy IV SP zmienia się charakter zajęć z informatyki, które stają się bardziej formalne. Uczniowie w dalszym ciągu pracują nad sytuacjami problemowymi przedstawianymi w sposób opisowy, w tym za pomocą ilustracji i historyjek. Powinni tworzyć je już jednak samodzielnie i potrafić wyciągnąć na tyle ogólne wnioski, żeby wynikające z nich działania złożyły się na własne realizacje w postaci programów lub czynności wykonywanych w innych programach.

W ten sposób uczniowie rozwijają podejście komputacyjne przy rozwiązywaniu różnorodnych sytuacji problemowych z różnych dziedzin. Posługiwanie się komputerem rozwija u nich umiejętności komunikacyjne, a także zdolność wyrażania swoich myśli i ich prezentacji. Potrafią pracować indywidualnie, a także zespołowo, w tym przy realizacji projektów dotyczących problemów z różnych dziedzin. Sieć służy im jako źródło informacji przydatnych w rozwiązywaniu zadań i problemów. Uczniowie powinni doceniać rolę współpracy w rozwoju swojej wiedzy i umiejętności oraz postępować odpowiedzialnie i etycznie w środowisku komputerowo-sieciowym.

Również uczniowie, którzy zrealizowali przedmiot zajęcia komputerowe w klasach IV–VI zgodnie ze starą podstawą programową kształcenia ogólnego dla 6-letniej szkoły podstawowej, są od klasy VII wprowadzani do myślenia komputacyjnego. Poznają podstawowe pojęcia informatyczne i rozwiązują algorytmicznie wybrane problemy. Stawiają pierwsze kroki w wizualnym lub tekstowym języku programowania. Dotychczas zdobyte wiedza i umiejętności informatyczne są u nich rozwijane i poszerzane.

Z kolei uczniowie, którzy w klasach IV–VI zrealizują przedmiot informatyka zgodnie z nową podstawą programową kształcenia ogólnego dla 8-letniej szkoły podstawowej, zostaną wcześniej wprowadzeni do myślenia komputacyjnego. Poznają już bowiem wcześniej podstawowe pojęcia informatyczne i nauczą się rozwiązywać algorytmicznie wybrane problemy, programując przy tym ich rozwiązania. Dotychczas zdobyte przez nich wiedza i umiejętności informatyczne będą więc rozwijane i poszerzane, a uczniowie będą stawiać swoje pierwsze kroki w tekstowym języku programowania.

Przy użyciu dostępnego oprogramowania uczniowie powinni realizować projekty i rozwijać kompetencje zespołowego rozwiązywania problemów pochodzących z różnych dziedzin.

Aby te cele były osiągnięte, podczas zajęć każdy uczeń powinien mieć do swojej dyspozycji osobny komputer z dostępem do internetu i odpowiednim oprogramowaniem. W trakcie prac nad projektami (indywidualnymi lub zespołowymi) uczniowie powinni mieć również możliwość korzystania z komputerów lub innych urządzeń cyfrowych, w zależności od potrzeb wynikających z charakteru zajęć, realizowanych celów i tematów. Najlepiej byłoby zatem, aby w części dotyczącej programowania oraz przy realizacji innych zagadnień związanych z pracą przy komputerze edukacja informatyczna odbywała się w pracowni komputerowej przystosowanej do wieku uczniów.

Innym koniecznym warunkiem gwarantującym realizację powyższych założeń jest odpowiednie przygotowanie metodyczne i merytoryczne nauczyciela. Powinien on mieć wiedzę merytoryczną znacząco wykraczającą poza zakres podstawy programowej, a także umieć wykorzystać różnorodne metody pracy, porównywać zapisy algorytmów w różnych postaciach i wreszcie mobilizować uczniów do stosowania poznanych algorytmów w rozwiązywaniu problemów pokrewnych.

W niniejszym zeszycie przyglądamy się kilku popularnym narzędziom i rozwiązaniom, które ułatwiają nauczycielom informatyki realizację powyższych postulatów.

Wskazówki metodyczne

Środki dydaktyczne

Zeszyty 2 i 3 w obu zestawach materiałów poświęconych myśleniu komputacyjnemu przybliżają problematykę nauki programowania i doskonalenia myślenia algorytmicznego z wykorzystaniem środków dydaktycznych. Należą do nich np. klocki, gry i roboty. Przedstawiona w zeszytach charakterystyka poszczególnych rozwiązań, jak i poniższe przypomnienie funkcji, jakie pełnią środki dydaktyczne w procesie kształcenia, pomogą w odpowiednim ich doborze do celów lekcji.

Środki dydaktyczne są niezbędnym elementem procesu nauczania, w tym nauczania programowania. Ich główną funkcją jest wspieranie czynności nauczyciela oraz pomoc uczniom w lepszym zrozumieniu przekazywanych treści. Nie będzie przesadą stwierdzenie, że dzięki środkom dydaktycznym uczniowie mogą w większym stopniu odwołać się do swojej wyobraźni, pobudzić myślenie, czego efektem jest lepsze zapamiętywanie.

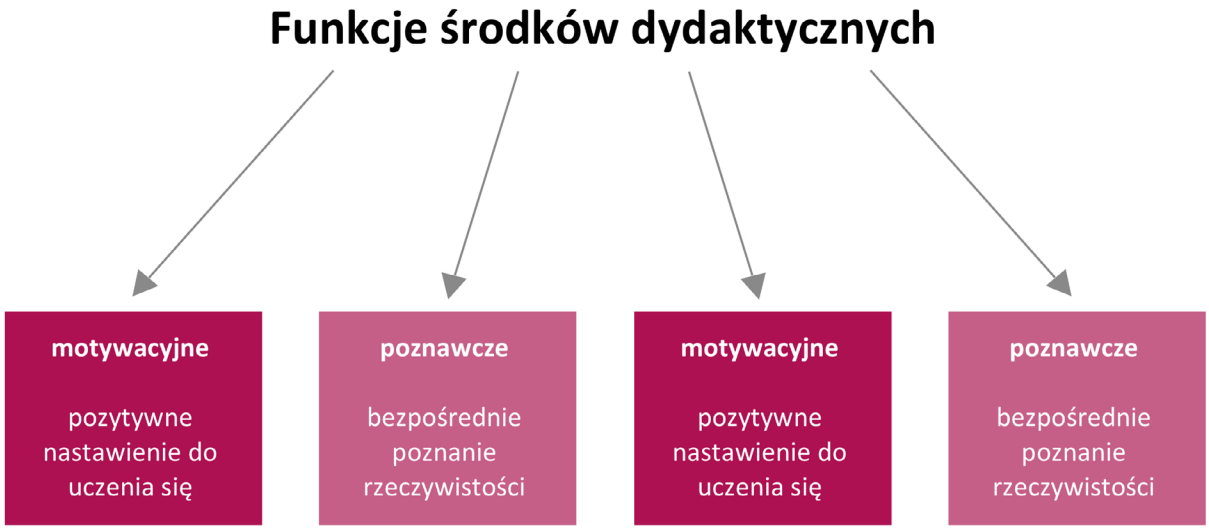
Franciszek Bereźnicki (2001) tak określił funkcje środków dydaktycznych:

„**Funkcja motywacyjna** polega na wywołaniu pozytywnego nastawienia do uczenia się przez budzenie zaciekawienia i zainteresowania dla przedmiotu poznania. Poprawnie wykonane pod względem pedagogicznym, technicznym i artystycznym środki dydaktyczne wywołują nie tylko określone przeżycia intelektualne, ale również wzruszenia, przeżycia emocjonalno-ekspresyjne, przez co rozbudzają zaangażowanie, zaciekawienie i zainteresowanie materiałem nauczania (...). Tak więc środki dydaktyczne, szczególnie audiowizualne, oddziałują na sferę emocjonalną, wyzwalają i wzmacniają motywy uczenia się, wywołują przeżycia i kształtują postawy uczniów.

Funkcja poznawcza polega na tym, że dzięki środkom uczący się poznaje bezpośrednio określoną rzeczywistość.

Funkcja kształcąca sprowadza się do rozwijania zdolności poznawczych (spostrzegawczości, wyobraźni, myślenia, pamięci) oraz kształtowania odpowiednich umiejętności i sprawności.

Funkcja dydaktyczna wyraża się w tym, że środki dydaktyczne są niejednokrotnie głównym źródłem wiadomości dla uczniów, ułatwiają ich zrozumienie, utrwalenie i sprawdzenie stopnia opanowania. Środki dydaktyczne mogą służyć ilustracji werbalnego przekazywania wiadomości, mogą pomagać uczniom w tworzeniu uogólnień, są nieodzowne w trakcie weryfikacji hipotez".



Rys. 1. Funkcje środków dydaktycznych wg F. Bereźnickiego

Powyższe funkcje środków dydaktycznych mogą się ze sobą łączyć i wzajemnie uzupełniać.

Według Wincentego Okonia (1996) środki dydaktyczne pełnią następujące funkcje:

- ułatwiają i pogłębiają poznawanie rzeczywistości,
- sprzyjają poznawaniu wiedzy o danej rzeczywistości,
- pomagają w kształtowaniu postaw i emocjonalnego stosunku do rzeczywistości,
- wspomagają rozwijanie działalności przekształcającej daną rzeczywistość.

Jak podaje Teresa Duda (b.r.), „obecnie środki dydaktyczne efektywnie wspomagają i współuczestniczą w procesie kształcenia, wspomagają realizację wszystkich jego ogniw, pełniąc następujące funkcje:

1. wprowadzającą i motywującą – której głównym celem jest organizowanie sytuacji problemowych, a co za tym idzie – przygotowanie uczniów do aktywnej pracy na zajęciach szkolnych;
2. źródłową i weryfikacyjną – środki dydaktyczne występują tu w postaci głównego źródła wiedzy, są materiałem do weryfikacji początkowych hipotez;
3. syntetyzującą i utrwalającą – środki dydaktyczne są tu traktowane jako pomoc w tworzeniu pewnych uogólnień, syntez i struktur wiedzy oraz utrwaleniu poznanego zakresu materiału;
4. zastosowawczą i wdrożeniową – chodzi tu o połączenie teorii z praktyką, pokazanie wzorów wykonania pewnych określonych czynności,

5. kontrolno-oceniającą”.

To, co wydaje się najważniejsze dla wszystkich przedstawionych tu funkcji środków dydaktycznych, to jest ich wspólna cecha polegająca na możliwości tworzenia różnorodnych sytuacji dydaktycznych, w których uczeń będzie rozwiązywał określony problem z pomocą dostępnych środków dydaktycznych.

Przygotowana przez nas charakterystyka wybranych środków dydaktycznych do nauki programowania, zaprezentowana w dalszej części niniejszego zeszytu, obejmuje m.in. te ich cechy, które pozwolą na uwzględnienie funkcji motywacyjnych, poznawczych, kształcących, dydaktycznych i wychowawczych, jakie powinny pełnić w procesie kształcenia.

Narzędzia do nauki programowania w procesie rozwoju myślenia komputacyjnego

Jak już pisaliśmy, myślenie komputacyjne to powtarzalny kilkustopniowy proces myślowy polegający na znajdowaniu rozwiązań dla złożonych problemów. Zajęcia z programowania w dużym stopniu polegają na przeprowadzeniu uczniów przez proces rozwiązywania określonego problemu, który staje się osią lekcji. Przypomnijmy schemat działania w sytuacji problemowej opisany w Zeszycie 1. Posłuży nam on w dalszej kolejności do refleksji nad doborem środków dydaktycznych wspomagających pracę ucznia w kolejnych etapach rozwiązywania problemów.

Prześledźmy ten proces, uzupełniając go również o wskazówki metodyczne dla nauczyciela dotyczące organizacji pracy na lekcji (oprac. na podst. *Uczymy dzieci…*, 2017).

1. Rozpatrywanie sytuacji problemowej.

    a) Rozpoznanie problemu w zależności od stopnia zaawansowania i wiedzy uczniów może się odbywać w formie pracy indywidualnej, grupowej lub podczas wspólnej dyskusji.
    b) Poproś uczniów o zastanowienie się nad problemem i zapisanie swoich myśli.
    c) Warto podsumować pracę indywidualną lub grupową na forum klasy, np. tworząc mapę myśli.
    d) Pamiętaj o słowach kluczowych, które pomogą uczniom we właściwym zdefiniowaniu problemu.

2. Poszukiwanie i określenie sposobu przekształcenia sytuacji problemowej w sytuację pożądaną.

    a) W poszukiwaniu różnych sposobów rozwiązań sytuacji problemowej pomogą uczniom pytania naprowadzające, np.

        - Czy spotkaliście się już wcześniej z podobnym problemem?
        - Jakie są możliwe sposoby rozwiązania problemu?

- Który ze sposobów będzie najbardziej efektywny, np. najszybszy, najkrótszy?
- Dlaczego wybraliście taki sposób?
- Co trzeba zrobić, żeby osiągnąć cel?
- Czy potrzebujecie dodatkowych danych, żeby rozwiązać problem?

b) Opracowanie algorytmu prowadzącego do rozwiązania sytuacji problemowej:

- jeśli uczniowie mieli już do czynienia z podobnym problemem, opracowanie algorytmu powinno się odbywać w formie pracy indywidualnej lub w parach;
- nauczyciel powinien kontrolować postępy pracy uczniów i w miarę potrzeby pomagać w pisaniu algorytmu;
- na tym etapie rozwiązywania problemu konieczna może się okazać indywidualizacja: uczniowie słabiej radzący sobie z tematem będą wymagali większej pomocy, z kolei uczniowie zdolniejsi mogą skończyć pracę nad algorytmem dużo wcześniej;
- pomyśl nad zaangażowaniem uczniów zdolniejszych w pomoc uczniom słabszym.

3. Analiza wybranej drogi przekształcenia sytuacji problemowej pod względem możliwych zdarzeń utrudniających rozwiązanie.

a) Sprawdzenie poprawności działania algorytmu, czyli otrzymanego wyniku lub osiągniętego celu, poza środowiskiem wizualnego programowania lub innym środowiskiem programistycznym (praca indywidualna lub praca w parach nadzorowana przez nauczyciela).
b) Tworzenie programu będącego realizacją opracowanego algorytmu w środowisku wizualnego programowania lub innym środowisku programistycznym (jw.).

4. Poddanie ewaluacji osiągniętego stanu (cele zamierzone a osiągnięte) oraz drogi, która doprowadziła do osiągnięcia celu.

a) Testowanie programu w środowisku wizualnego programowania lub innym środowisku programistycznym (praca indywidualna lub grupowa, nadzorowana przez nauczyciela).
b) Prezentacja projektu lub rozwiązania problemu (prezentacje indywidualne lub grupowe na forum klasy).

Każdy kolejny etap edukacyjny przynosi zarówno uczniom, jak i nauczycielom coraz większe wyzwania. Problemy i przykłady, z którymi przychodzi się im mierzyć, są coraz bardziej skomplikowane. Znaczącą zmianą jest przejście z wizualnego języka programowania (np. Scratch, Scottie Go!) do tekstowego języka programowania (np. Python), a także wstęp do konstruowania i programowania prostych robotów. Poniższa prezentacja komercyjnych i niekomercyjnych narzędzi – środków dydaktycznych, została zorganizowana zgodne z tym porządkiem.

Przegląd narzędzi do programowania wizualnego

Scottie Go!

Zestaw do programowania Scottie Go!

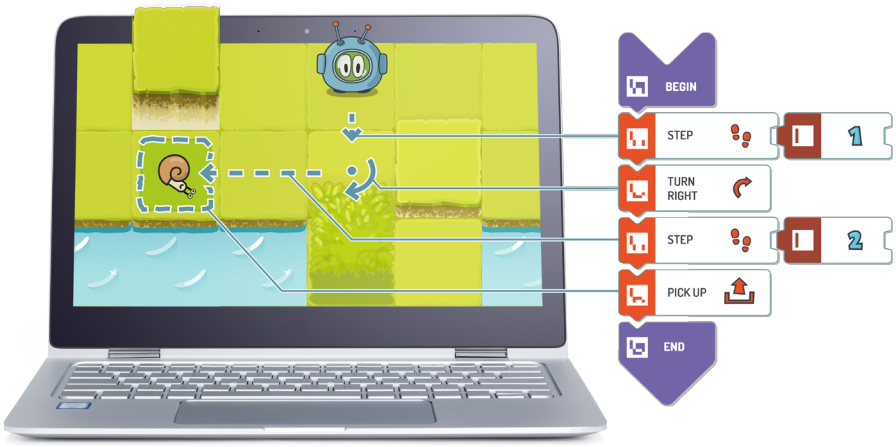
Scottie Go! jest grą edukacyjną z interaktywnym kursem programowania. To połączenie aplikacji edukacyjnej z blisko stu zadaniami o rosnącym poziomie trudności oraz kartonowych, rozpoznawanych przez aplikację klocków służących do pisania programów.

Gra doskonali umiejętności analitycznego i logicznego myślenia, uczy rozwiązywania skomplikowanych problemów i pracy w grupie, rozwija intuicję algorytmiczną.

Zestaw zawiera 179 klocków do nauki programowania, 91 zadań podzielonych na 10 modułów. Koncepty zawarte w grze to m.in. składanie liczb z cyfr, dodawanie, odejmowanie, układanie algorytmów, sterowanie postaciami na scenie, parametr, pętla, wyrażenia warunkowe, zmienna, funkcja.

Jak to działa?

Użytkownik za pomocą kartonowych klocków steruje kosmitą Scottiem. Wypełniając misję sprowadzenia przybysza z innej planety do domu, dziecko poznaje podstawowe pojęcia programowania. Kolejne zadania prezentują coraz wyższy poziom trudności.



Rys. 2. Przykład układania prostych komend w grze Scottie Go!

Gra w wersji edukacyjnej zapewnia dostęp do:

- bazy materiałów i rozwiązań metodycznych,
- plansz z zadaniami, które mogą posłużyć do opracowania kart pracy lub wyświetlania na ekranie podczas zajęć;
- konta dla nauczyciela gwarantującego dostęp do odblokowanych zadań we wszystkich modułach.

Gra Scottie Go! jest wspomagana aplikacją dostępną na tablet.

Podsumowanie

Scottie Go! to propozycja dla nauczycieli nauczania początkowego, ale może być również wykorzystana w klasie czwartej i piątej szkoły podstawowej. Gra ułatwia wyrównanie poziomu grupy w zakresie podstaw programowania wizualnego. Jest dobrym wstępem do kodowania w bardziej zaawansowanych środowiskach, a ponieważ wciąga do zabawy uczniów bez względu na ich poziom umiejętności, ułatwia pracę ze zróżnicowaną klasą. Grę wzbogaca baza materiałów metodycznych, np. gotowych pomysłów na lekcje. Scottie Go! został opracowany w Polsce i stanowi ciekawe uzupełnienie bogatej oferty narzędzi do programowania wizualnego dla dzieci.

Gra dostępna jest w różnych wariantach cenowych, zależnych od liczby kompletów i opcji dodatkowych. Cena podstawowego zestawu edukacyjnego bez dodatków to ok. 180 zł.

Przykładowy scenariusz zajęć z wykorzystaniem gry Scottie Go!

Poniższy scenariusz lekcji opracowany został przez Z. Karwasińskiego i M. Plebańską, metodyków Poznańskiego Centrum Superkomputerowo-Sieciowego. Mimo że materiał ten został zaprojektowany z przeznaczeniem dla I etapu edukacyjnego, równie dobrze sprawdzi się na początku II etapu, jeśli nauczyciel chce zrealizować zajęcia powtórkowe lub wyrównać umiejętności z zakresu podstaw programowania.
Zaproponowana struktura zajęć uwzględnia elementy procesu rozwiązywania problemu, o którym piszemy we wstępie i wskazówkach metodycznych, tj.

1. Rozpatrywanie sytuacji problemowej.
2. Poszukiwanie i określenie sposobu przekształcenia sytuacji problemowej w sytuację pożądaną.
3. Analiza wybranej drogi przekształcenia sytuacji problemowej pod względem możliwych zdarzeń utrudniających rozwiązanie.
4. Poddanie wielopodmiotowej ewaluacji osiągniętego stanu (cele wyobrażone a osiągnięte) oraz drogi, która doprowadziła do osiągnięcia celu.

Scenariusz lekcji a konspekt

Scenariusz lekcji jest, podobnie jak konspekt, zapisem planu projektowanych zajęć. Sama nazwa wskazuje na rozbudowaną formę planu, który będzie w tym wypadku zawierał takie szczegóły, jak: pytania do uczniów i przewidywane odpowiedzi, dokładne opisy przebiegu lekcji oraz metod i technik pracy, a także szczegółowej ewaluacji itp. Nauczyciele w swojej praktyce zawodowej korzystają z różnych źródeł metodycznych, w których napotykają na różnorodne formy zapisu planu zajęć, od mniej szczegółowych (konspektów), sygnalizujących jedynie temat, cele, metody, przebieg lekcji oraz sposób ewaluacji, po bardzo rozbudowane scenariusze będące dokładnymi zapisami lekcji. To, z jakich form będą korzystali, zależy wyłącznie od ich potrzeb na danym etapie rozwoju zawodowego.

Tytuł zajęć: Akcja ratunkowa. Sekwencje zdarzeń, instrukcje.

Czas trwania: 90 minut

Cel główny:
Zapoznanie uczniów z pojęciem i zasadami tworzenia zrozumiałych instrukcji

Cele szczegółowe:
Uczeń po zakończonych zajęciach:

1. wyjaśnia swoimi słowami, czym jest instrukcja;
2. wyjaśnia, w jaki sposób należy tworzyć instrukcje, żeby były zrozumiałe dla kolegi, rodzica, komputera;
3. używając prostych poleceń, potrafi zaprogramować robota, obiekt na ekranie, aby pokonał wyznaczoną trasę;
4. podaje prosty przykład instrukcji dla codziennych czynności domowych i szkolnych, współpracuje w zespole tworzącym program.

Cele niespecyficzne:

1. Uczniowie przekonują się, jak ważne jest jasne formułowanie poleceń, uczą się jasno komunikować swoje myśli.
2. Pracując w grupach, uczniowie ćwiczą umiejętność dyskusji i negocjacji.

Wymagania technologiczne, jakie musi spełnić szkoła/sala/przestrzeń dydaktyczna, w której odbywają się zajęcia, potrzebne materiały:

- minimum jeden tablet o parametrach pozwalających na uruchomienie aplikacji mobilnej Scottie Go!, łącze internetowe,
- aplikacja mobilna Scottie Go!,
- zestawy Scottie Go! – jeden na 2–3 osoby,
- arkusze papieru z podziałem na 6 pól,
- karty pracy,
- ołówki, kredki, nożyczki, klej.

Metody pracy

- pogadanka
- pantomima
- gra edukacyjna

Ogólny zarys

Podczas zajęć uczniowie poznają pojęcie instrukcji oraz zasady tworzenia dobrych instrukcji. Wykonując zadania manualne, ruchowe oraz z wykorzystaniem klocków i aplikacji, uczą się tworzyć jasne i zrozumiałe komunikaty, które pozwolą zrozumieć, jak działa i „rozumie" komputer. Zadania zespołowe dodatkowo pozwolą im poprawić umiejętności pracy zespołowej i komunikacji.

Przebieg zajęć

Część wstępna

Cele lekcji (1 minuta)
Przedstaw uczniom, jak będzie wyglądała lekcja. Określ czas lekcji oraz narzędzia, których będą używać.

Czym jest instrukcja? – pogadanka (5 minut)
Zapytaj uczniów, z czym kojarzy im się słowo instrukcja. Stymuluj i motywuj ich do podawania własnych przykładów. Zapytaj także o to, kto i w jakich okolicznościach wydaje instrukcje (polecenia).

Wykorzystaj pytania:

- Co to jest instrukcja?
- Kto w waszych domach najczęściej wydaje instrukcje?
- Podajcie przykłady takich instrukcji.
- Czy wykonywanie instrukcji jest łatwe, czy trudne?
- Czy w instrukcjach ważna jest kolejność wykonywanych zadań?
- Czy możecie wskazać miejsca, sytuacje życiowe, gdzie również trzeba stosować się do instrukcji?
- Dlaczego ważne jest stosowanie się do zasad zawartych w instrukcji?

Kieruj rozmową tak, aby uzyskać informację o tym, że instrukcja to zbiór poleceń, które określają co, w jaki sposób, a często nawet w jakiej kolejności powinno być wykonywane. Instrukcja musi określać listę czynności w taki sposób, aby osoba, która otrzyma instrukcję, wykonała zadanie bezbłędnie.

Uczniowie mogą dojść do wniosku, że:

- Instrukcje często wydają rodzice, mówiąc, w jaki sposób umyć zęby, o której trzeba rano wstać do szkoły, w co i jak się ubrać itp.
- Słuchanie zasad zapisanych w instrukcji bywa trudne, bo jest wiele poleceń, które trzeba wykonać.

Postaraj się uzyskać informację od uczniów, że instrukcję wykonują: pracownicy w firmach – wykonują polecenia kierowników, żołnierze – wykonują rozkazy, ratownicy – stosują procedury medyczne.

Pytania dobieraj do wieku, kompetencji i sytuacji (np. życiowej) uczniów.

Część zasadnicza

Przygotowanie herbaty – pantomima (5 minut)

Zaproponuj uczniom zabawę w programowanie robota. Robotem jesteś ty i uczniowie mają wymyślić dla ciebie instrukcję przygotowania herbaty. Ty jako robot nie mówisz, więc posługuj się gestem i mową ciała zamiast słowami i eksponatami.

Pamiętaj, że jako robot nie rozumiesz trudnych zadań. Staraj się zmusić uczniów do wydawania prostych i krótkich poleceń. Wprowadź nieco humoru, przekręcając pewne czynności, gdy te będą niejasne, niekonkretne lub będą wydawane w niewłaściwej kolejności. Przykładowo, uczniowie mówią: „Nalej wody do szklanki” (woda znajduje się w kranie, o czym ci nie powiedzieli, ani o tym, gdzie jest szklanka, więc nabierasz wodę do rąk itp.). W młodszej grupie możesz podać inny przykład, np. „Pokaż, co musisz zrobić, kiedy się obudzisz, zanim pójdziesz do szkoły”.

Mycie zębów – pierwsza instrukcja (10 minut)

Rozdaj uczniom materiały do rysowania, tj. arkusze papieru podzielone na 6 części i kredki. Zaproponuj stworzenie obrazkowej instrukcji dla Scottiego, żeby nauczyć go mycia zębów.

Poproś uczniów, aby przed rozpoczęciem rysowania przypomnieli sobie, z jakich etapów składa się mycie zębów i dokładnie zaplanowali, co w którym okienku będzie narysowane. W trakcie wykonywania rysunków zwróć uwagę na poprawną kolejność w historyjce.

Po skończonej pracy uczniowie tną historyjki na 6 części, tasują karty i wymieniają się taliami z historyjkami.

Mycie zębów – posortuj historyjkę (5 minut)

Drugim zadaniem uczniów jest posortowanie historyjki kolegi i ułożenie jej we właściwej kolejności.

Przygody Scottiego – wprowadzenie do cyklu lekcji (5 minut)

Pokaż uczniom pudełko gry Scottie Go!. Zapytaj, kim jest postać na pudełku i opowiedz historię Scottiego. Oto przykładowa opowieść:

„Kto mógł przypuszczać, że losy sympatycznego kosmity Scottiego będą zależeć od ciebie i twoich umiejętności programowania? Jest rok 2030. Pojazd Scottiego uległ awarii i kosmita musiał przymusowo lądować na naszej planecie. Niestety, podczas lądowania uszkodził pojazd jeszcze bardziej. Scottie nie wie, gdzie się znajduje, ale udało mu się wezwać pomoc. Satelity wysłały nam jedynie niewyraźne zdjęcie, a dotarcie do pojazdu kosmicznego nie jest łatwe. Na razie może tam dotrzeć tylko robot."

Mapa – misja ratunkowa (10 minut)

Podczas tego zadania uczniowie muszą napisać program sterujący robotem.

Przygotowanie: Rozdaj karty pracy, kartę pomocniczą z żetonami/instrukcjami oraz nożyczki i klej. Poproś uczniów o wycięcie żetonów.

Zadanie: Po przygotowaniu żetonów poproś ucznia o przeczytanie lub przeczytaj sam polecenie na karcie pracy. Zadanie polega na napisaniu za pomocą żetonów prostej instrukcji dla zespołu ratunkowego, aby jak najszybciej dotarł do rozbitego statku Scottiego.

Uczniowie przygotowują instrukcję, przyklejając na karcie pracy żetony w odpowiedniej kolejności.

Dostępne żetony: Start, Krok, W lewo, W prawo, Koniec.

Karta pracy zawiera mapę stworzoną na planszy zbliżonej do szachownicy. Na jednym z pól znajduje się wrak statku Scottiego, na innych punkt startowy oraz wiele dodatkowych przeszkód. Uczniowie mogą dotrzeć do celu na kilka sposobów, ale najlepsza droga to ta z najkrótszą instrukcją.

Scottie Go! – pierwsza misja (5 minut)

Rozdaj pudełka Scottie Go! oraz karty z zadaniami. Poproś uczniów o przygotowanie stanowisk do pracy i rozpakowanie pudełek. Wykonajcie pierwszą misję Scottiego wspólnie. Po ułożeniu programu przejdź po sali, aby zeskanować kody i sprawdzić poprawność napisanych programów.

Uczniowie wyciągają poszczególne klocki i układają je na planszy, a następnie dokładają kolejne klocki.

Karty pracy z planszami mogą zostać wyświetlone na ekranie w taki sposób, aby każdy uczeń widział co najmniej trzy kolejne zadania.

Scottie Go! – instrukcje proste (40 minut)

Poproś uczniów o wykonanie 10 zadań z pierwszego modułu gry Scottie Go!. Skanuj kody, gdy zespół zgłosi gotowość i chęć przejścia do kolejnego etapu.

Chodzenie do tyłu jest jedną z nowych umiejętności Scottiego. W grze realizowane jest to przez użycie znaku <MINUS> przed wartością określającą liczbę kroków kosmity.

Uczniowie, którzy na lekcjach matematyki nie spotkali się jeszcze z liczbami ujemnymi, bardzo szybko i skutecznie stosują znak <MINUS>, gdy zauważą podobieństwo do dodatnich i ujemnych temperatur, które są dla nich już oczywiste i stosowane.

Część końcowa

Podsumowanie lekcji (3 minuty)

Zapytaj, czy uczniowie pamiętają, co to jest instrukcja i jakie ma cechy. Przypomnij uczniom, co się może wydarzyć, gdy instrukcje będą niejasne i wydawane w złej kolejności. Zapytaj, czy łatwiej tworzy się programy samodzielnie, czy współpracując w zespole.

Dodatkowe zadanie:
Uporządkuj historię Scottiego.

Oczekiwane efekty pracy

- Definiowanie problemu lub sytuacji problemowej samodzielnie lub w grupie.
- Analiza problemu lub sytuacji problemowej.
- Szukanie różnych dróg rozwiązań.
- Wybór najefektywniejszej (np. najszybszej, najkrótszej) drogi rozwiązania problemu.
- Opracowanie algorytmu prowadzącego do rozwiązania problemu.
- Sprawdzenie poprawności działania opracowanego algorytmu (czyli otrzymania zakładanego wyniku lub osiągnięcia celu) poza środowiskiem wizualnego programowania lub innym środowiskiem programistycznym.
- Tworzenie programu będącego realizacją opracowanego algorytmu w środowisku wizualnego programowania lub innym środowisku programistycznym.
- Testowanie programu w środowisku wizualnego programowania lub innym środowisku programistycznym.
- Prezentacja rozwiązania problemu.

Ewaluacja efektów zajęć z wykorzystaniem TIK

Ewaluacja efektów takich zajęć powinna polegać na obserwacji aktywności uczniów na poszczególnych etapach realizowanej lekcji, w szczególności w zakresie sposobu przeprowadzania analizy problemów, szukania różnych rozwiązań, wyboru najefektywniejszej drogi rozwiązania problemu oraz testowania.

Uwagi

Należy zwrócić uwagę na konieczność dostosowania wymagań oraz form i metod pracy do indywidualnych potrzeb uczniów, w tym uczniów ze specjalnymi potrzebami edukacyjnymi, a także do ich doświadczeń, możliwości językowych i psychomotorycznych.

Zadanie dla nauczyciela stażysty

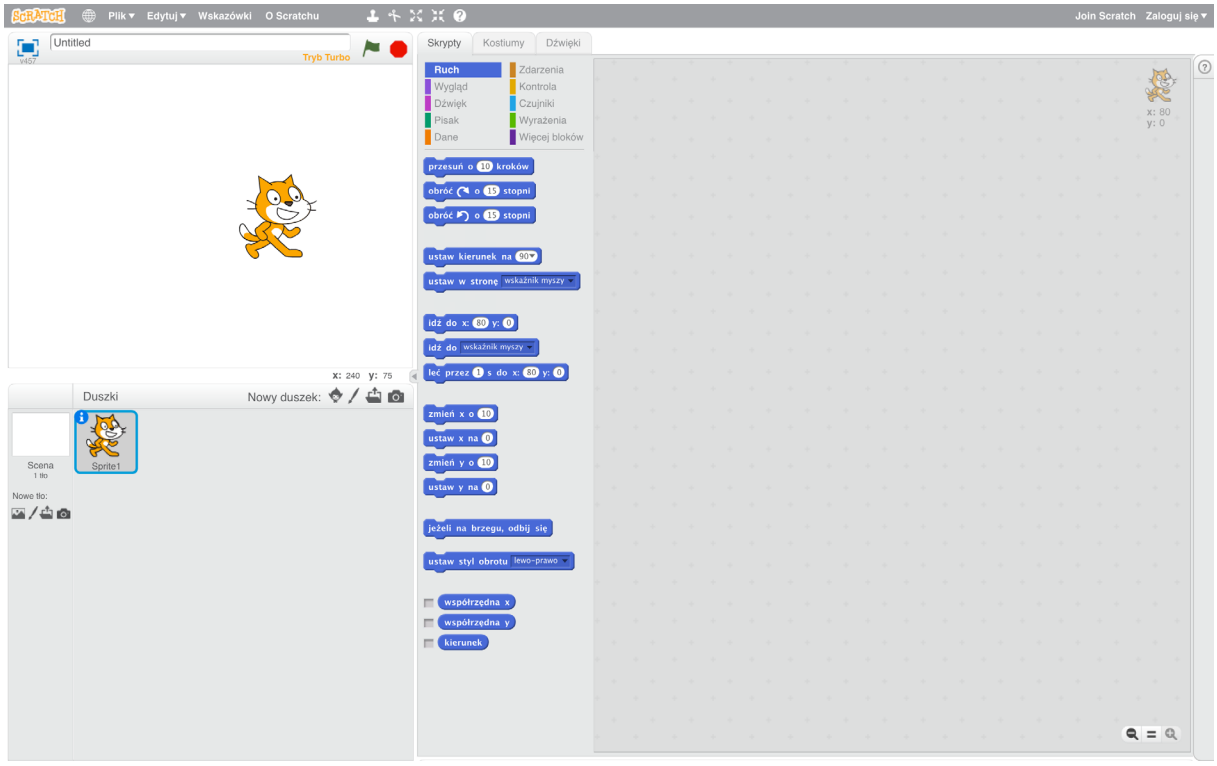
Proces myślenia komputacyjnego składa się z następujących etapów:

1. Dekompozycja: sformułowanie problemu i rozłożenie go na części składowe.
2. Analiza: rozpoznanie prawidłowości właściwych problemowi.
3. Abstrahowanie: eliminowanie elementów nieistotnych przez uogólnianie.
4. Tworzenie algorytmu: rozwiązanie problemu.

Aby nauczyć się rozpoznawać etapy myślenia komputacyjnego w przebiegu lekcji i działaniach uczniów, przeczytaj zamieszczony w niniejszym zeszycie scenariusz lekcji „Akcja ratunkowa" z wykorzystaniem gry Scottie Go!. Wskaż fragmenty, które twoim zdaniem odnoszą się bezpośrednio do wymienionych wyżej elementów.

Scratch

Jest to interpretowany wizualny język programowania stworzony przez Mitchela Resnicka w Massachusetts Institute of Technology Media Lab. Powstał w celach edukacyjnych jako środek do nauczania podstaw programowania. Istotne jest, że program ten jest bezpłatny i dostosowany do wszystkich popularnych systemów operacyjnych.



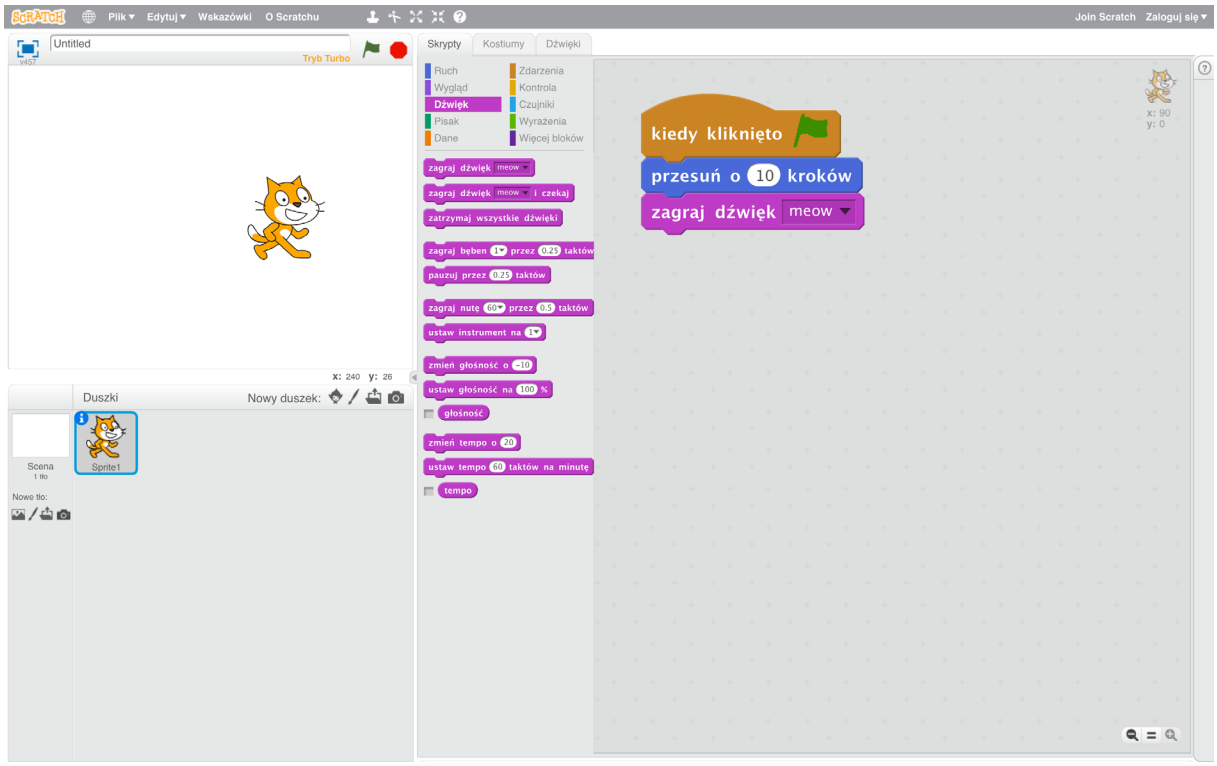
Wygląd interfejsu programu Scratch.

Scratch umożliwia tworzenie interaktywnych historyjek, gier, animacji, a nawet muzyki. Zasady programowania polegają na przeciąganiu i układaniu elementów języka w kształcie puzzli. Połączone ze sobą elementy tworzą kod dla wybranego obiektu. Program ma obszerną

bibliotekę obiektów. Można również importować obiekty z zewnątrz lub tworzyć je z poziomu programu dostępnego w edytorze grafiki.

Jak to działa?

Największe pole po prawej stronie to tzw. strefa skryptów, do której przenosimy bloki z paska na środku ekranu.



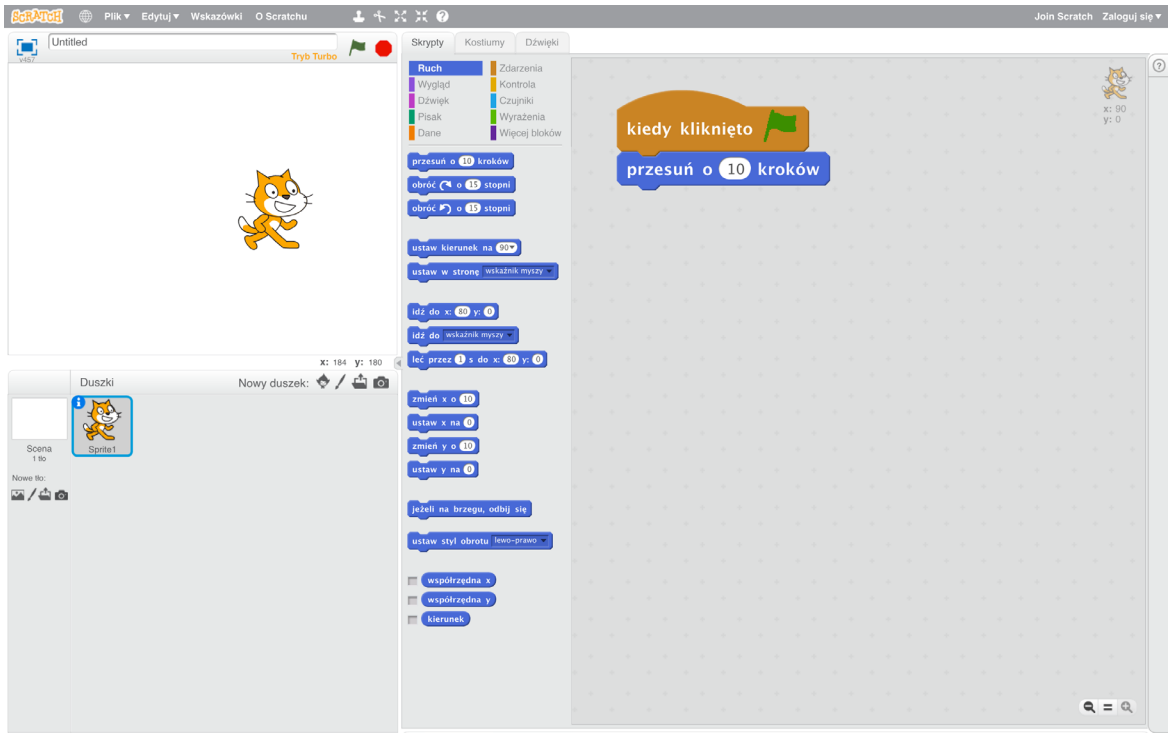
Interfejs programu Scratch z otwartym oknem do pisania skryptów.

Ułożony przez nas skrypt steruje postacią widoczną w oknie po lewej stronie ekranu. To okno podglądu, na którym widzimy efekt naszego „kodowania”.

Aby rozpocząć kodowanie, należy przeciągnąć blok do strefy skryptów. Pierwszy blok ma za zadanie uruchomić cały skrypt. Po kliknięciu zielonej chorągiewki (nad oknem podglądu) skrypt zostanie uruchomiony.

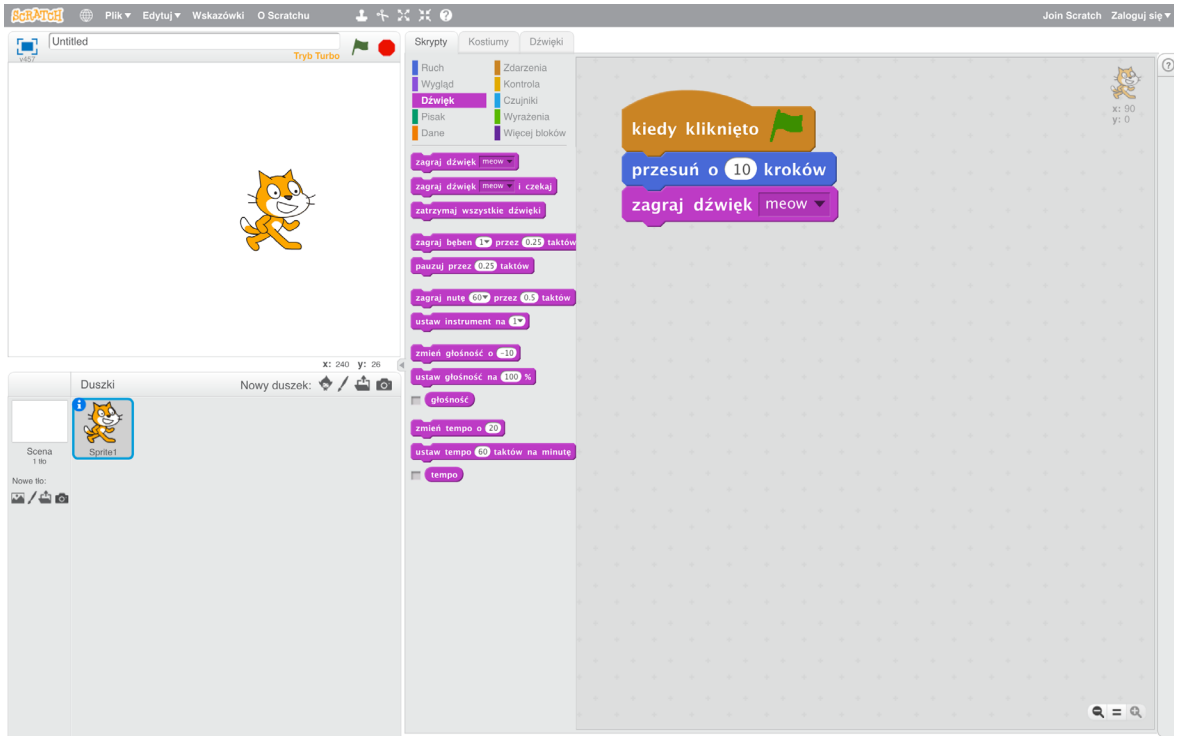
Przykład

Dodajemy blok odpowiadający za ruch postaci.



Tworzenie kodu w programie Scratch: przesunięcie postaci.

Teraz po kliknięciu zielonej chorągiewki postać przesunie się o 10 kroków.
Dodając bloki, możemy „zaprogramować” wiele różnych akcji, np. dodać dźwięk.



Tworzenie kodu w programie Scratch: dodawanie dźwięku.

W dołączonym do niniejszego opracowania filmie prezentujemy powyższą procedurę.

Użytkownicy programu Scratch mogą opracować wiele ciekawych akcji, animacji, rysunków, a nawet tworzyć dźwięki i muzykę. Program może być używany do programowania robotów, o czym piszemy w Zeszycie 3.

Podsumowanie

Scratch jest programem popularnym i bezpłatnym. Wystarczy wejść na stronę aplikacji i wybrać opcję „Wypróbuj". Dostępna jest też wersja offline. Programowanie w Scratchu stanowi doskonałe wprowadzenie do kodowania w bardziej zaawansowanych językach. Program umożliwia szybkie projektowanie gier, animacji, rysunków, komponowanie muzyki, co ułatwia nauczycielom konstruowanie konspektów lekcji z atrakcyjnymi celami i motywującymi, możliwymi do osiągnięcia podczas jednych zajęć rezultatami. Scratch wciąga uczniów do zabawy niezależnie od ich poziomu umiejętności, co ułatwia pracę ze zróżnicowaną klasą. Uczniowie zdolni mogą rozwiązywać trudniejsze problemy lub wspierać innych uczniów.

Strona programu zawiera wiele wskazówek i gotowych projektów, które można przeanalizować lub zmodyfikować. W internecie można łatwo znaleźć szeroką bazę materiałów metodycznych, samouczków i instrukcji do programu.

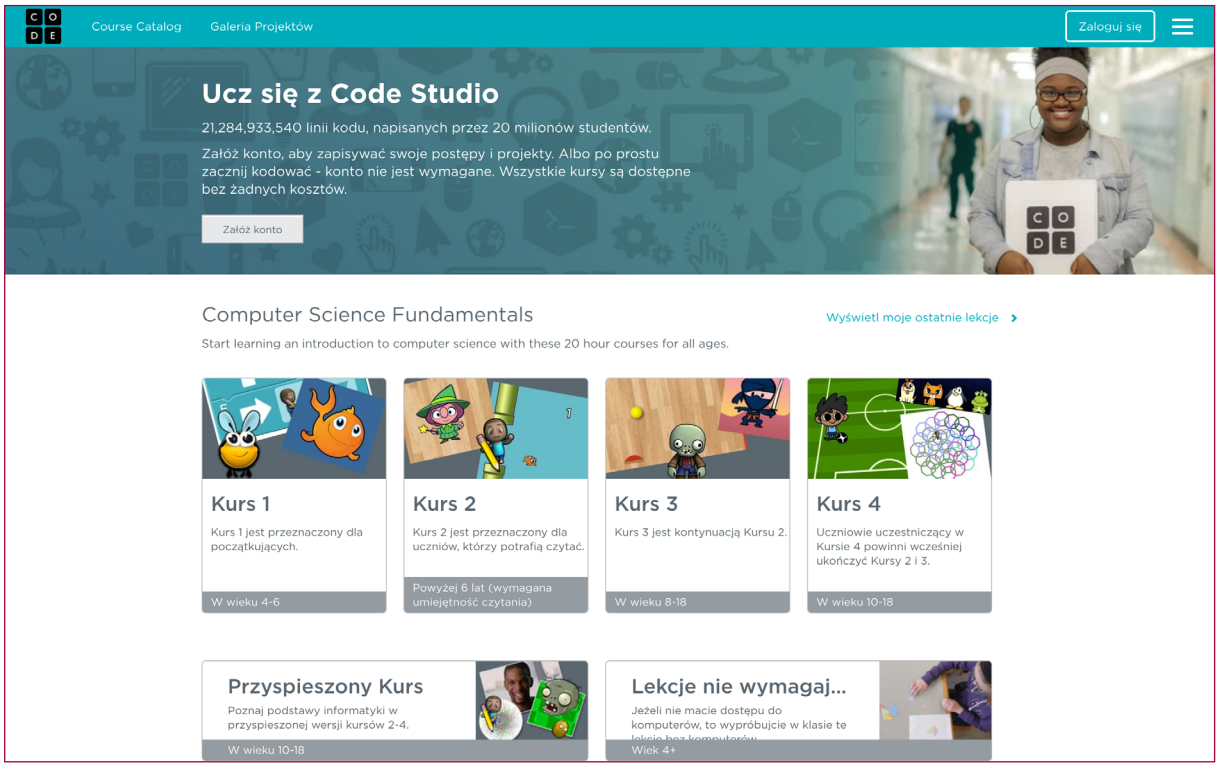
Zadanie dla nauczyciela stażysty

Interesującym źródłem pomysłów na lekcje i instrukcji obsługi programu Scratch jest strona internetowa Mistrzowie kodowania.

Wybierz jeden ze scenariuszy zajęć, który chciałbyś zaadaptować lub wykorzystać w inny sposób na swojej lekcji. Odnajdź i omów z mentorem elementy zawarte we wskazówkach metodycznych, tj. formułowanie celów, przebieg lekcji, ocenianie pod kątem wybranego scenariusza.

Code.org

Alternatywą dla programu Scratch jest system dostępny pod adresem Studio.code.org. Może on być szczególnie atrakcyjny dla dzieci, bo oferuje kursy programowania z ulubionymi bohaterami popularnych gier i kreskówek. Naszym zdaniem jest także jednym z najlepszych rozwiązań do nauki programowania wizualnego dla nauczycieli, którzy chcą rozwinąć swój warsztat.



Wygląd strony Studio.code.org z aktywnymi kursami programowania.

Portal proponuje gotowe zestawy pomysłów na gry i aplikacje dostosowane do odpowiednich grup wiekowych. Zawiera także przemyślane i skuteczne kursy, które można traktować jak gotowe rozwiązania do zastosowania w szkole i w domu (np. w ramach nauczania wyprzedzającego).

Jak to działa?

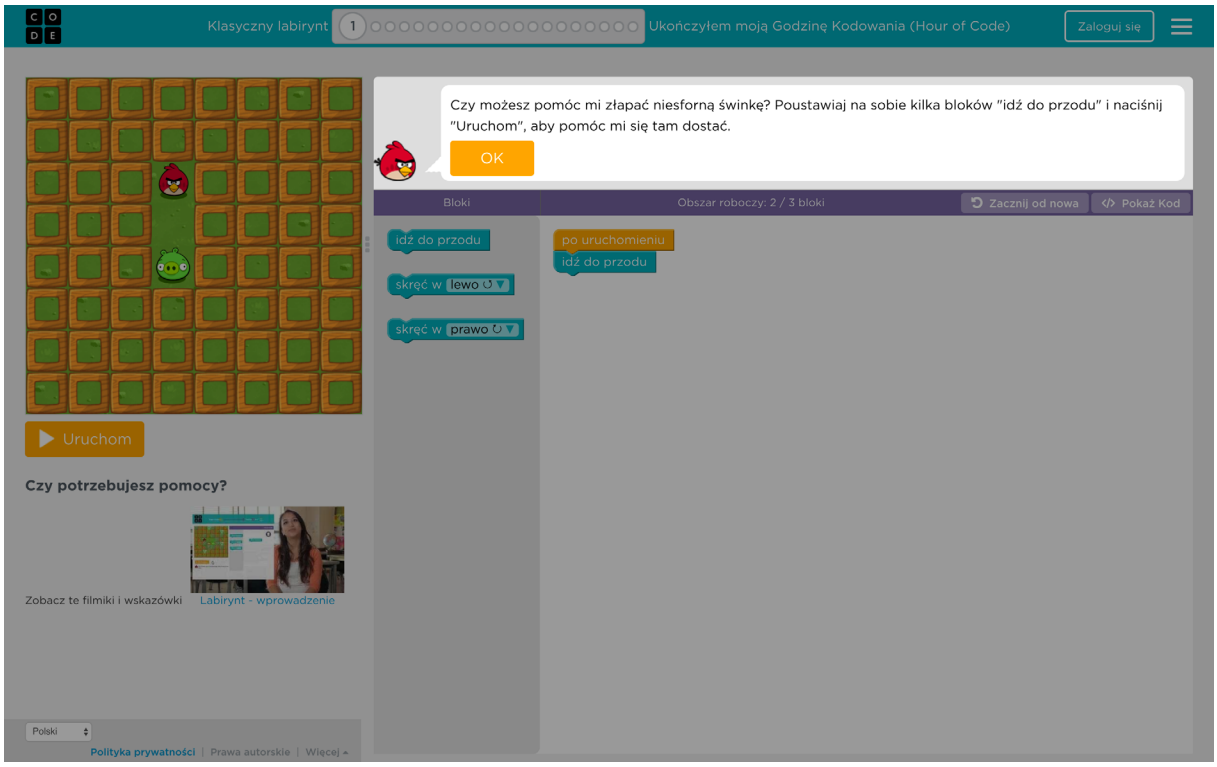
Programowanie w Studio.code.org przebiega analogicznie do programu Scratch.

Przykłady

Z naszych doświadczeń wynika, że najłatwiejszą drogą do zapoznania się z metodą programowania w Studio.code.org jest wykonanie poniższych ćwiczeń (w dowolnej kolejności).

Animuj bohaterów gry Angry Birds

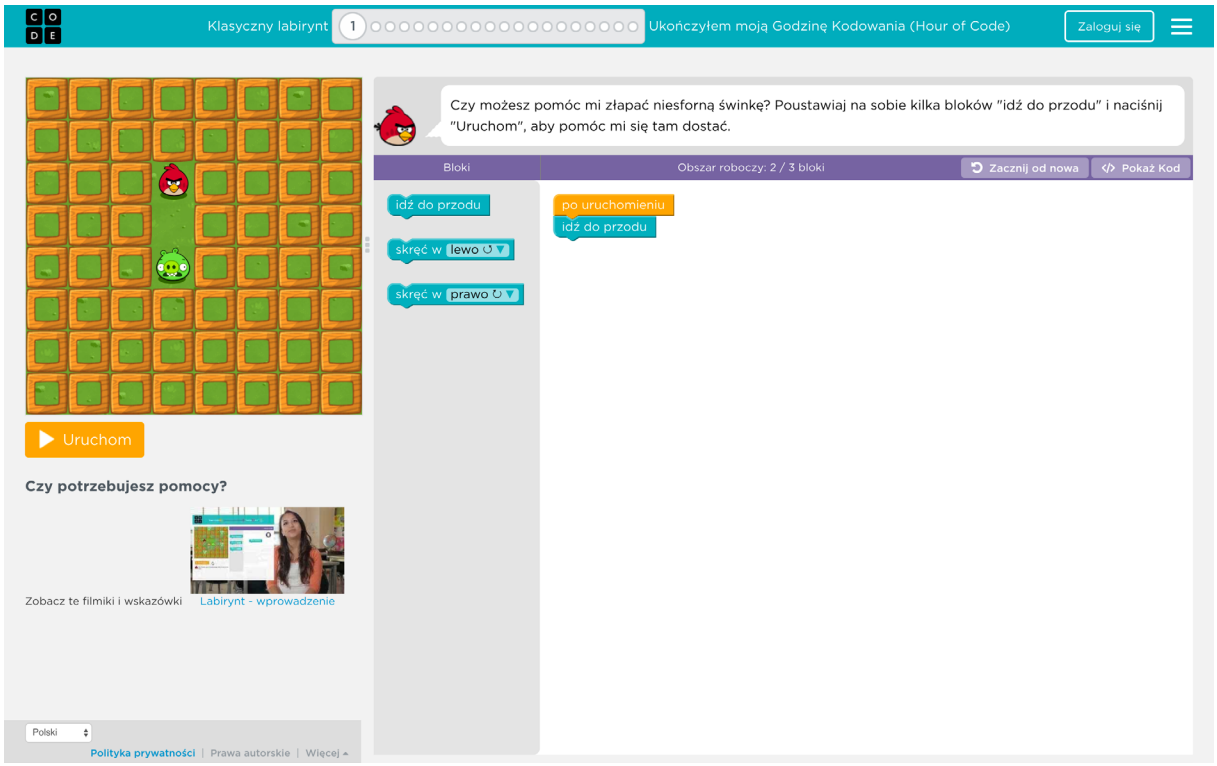
Lekcja dostępna pod adresem: https://studio.code.org/hoc/1.



Wygląd interfejsu przykładowej lekcji programowania na stronie Studio.code.org.

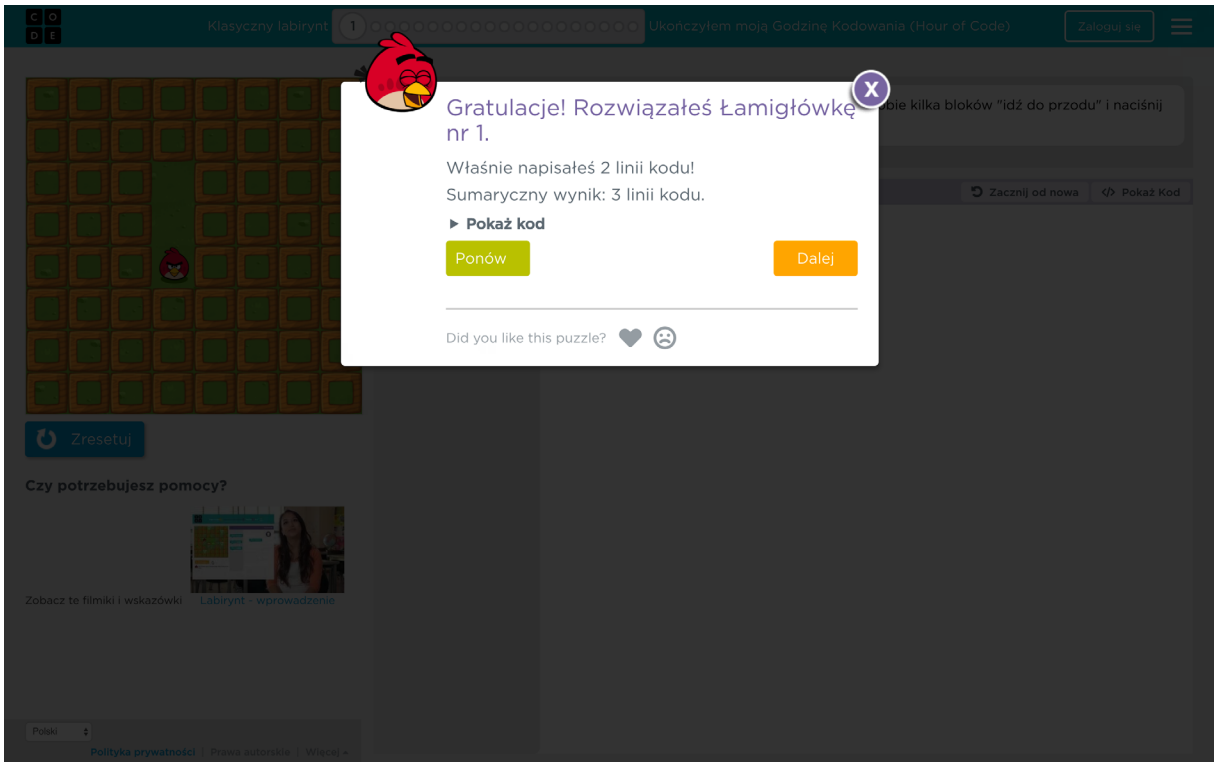
Nasze zadanie polega na zbudowaniu skryptu, który pozwoli czerwonemu ptaszkowi dojść do zielonej świnki. W obszarze roboczym mamy już dodane dwa bloki. Po kliknięciu przycisku Uruchom zobaczymy, jak zadziałają.

Program sam podpowiada nauczycielowi (i uczniowi równocześnie), jak zdefiniować problem, który można analizować podczas zajęć: „Czy możesz pomóc mi złapać niesforną świnkę?”, a także propozycje rozwiązania: „Poustawiaj na sobie kilka bloków »idź do przodu« i naciśnij »Uruchom«, aby pomóc mi się tam dostać”. Warto zwrócić uwagę na sposób projektowania funkcji motywacyjnej: występuje znany i lubiany bohater, obecne jest odwołanie do potrzeby pomagania, mamy obietnicę nagrody, którą jest możliwość samodzielnego napisania programu i prawidłowego rozwiązania postawionego problemu itp.



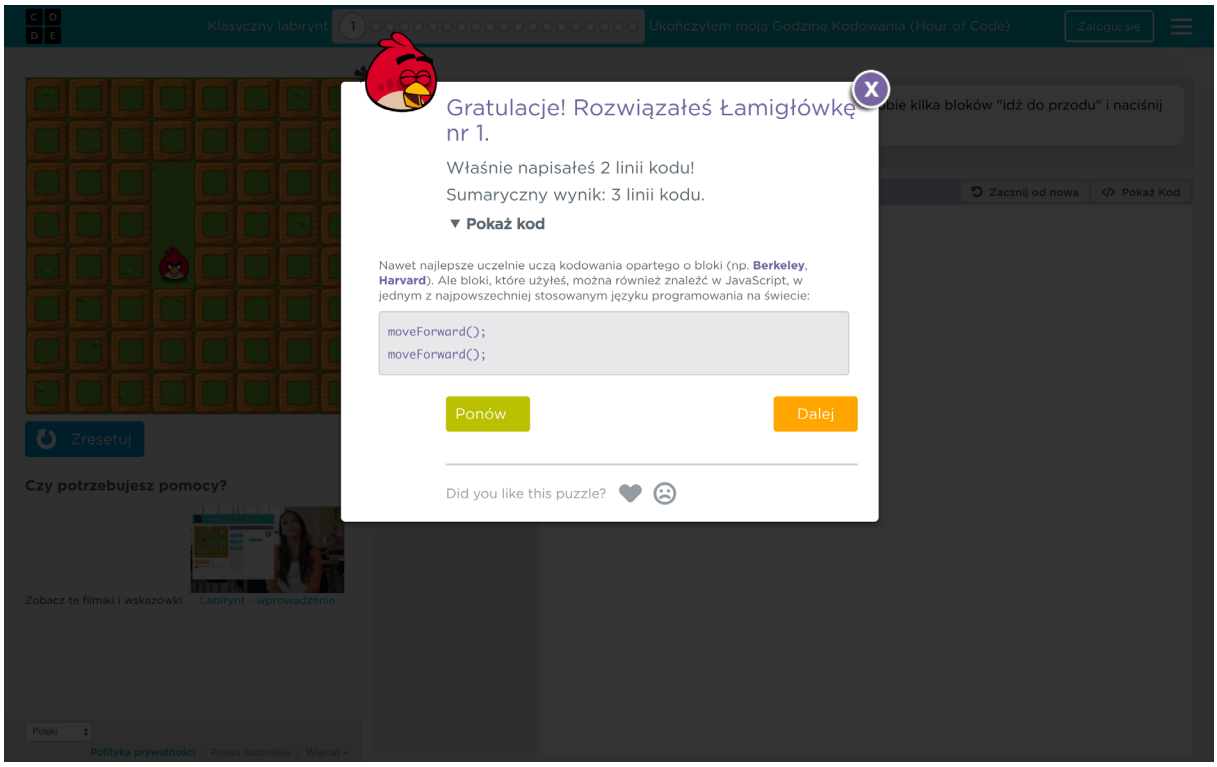
Pisanie skryptu na Studio.code.org: poruszanie postaci.

Widzimy, że wstawienie tylko jednego bloczka nie wystarczy. Program wyświetla użytkownikowi podpowiedzi. Jeśli dodamy kolejny element, uzyskamy oczekiwany efekt.



Komunikat o osiągnięciach generowany przez program Studio.code.org.

Program umożliwia wyświetlenie kodu, co ułatwi prowadzenie zajęć wprowadzających do bardziej zaawansowanych języków programowania. Rozwija także znajomość języka angielskiego, co jest istotne z punktu widzenia rozwijania kompetencji kluczowych.

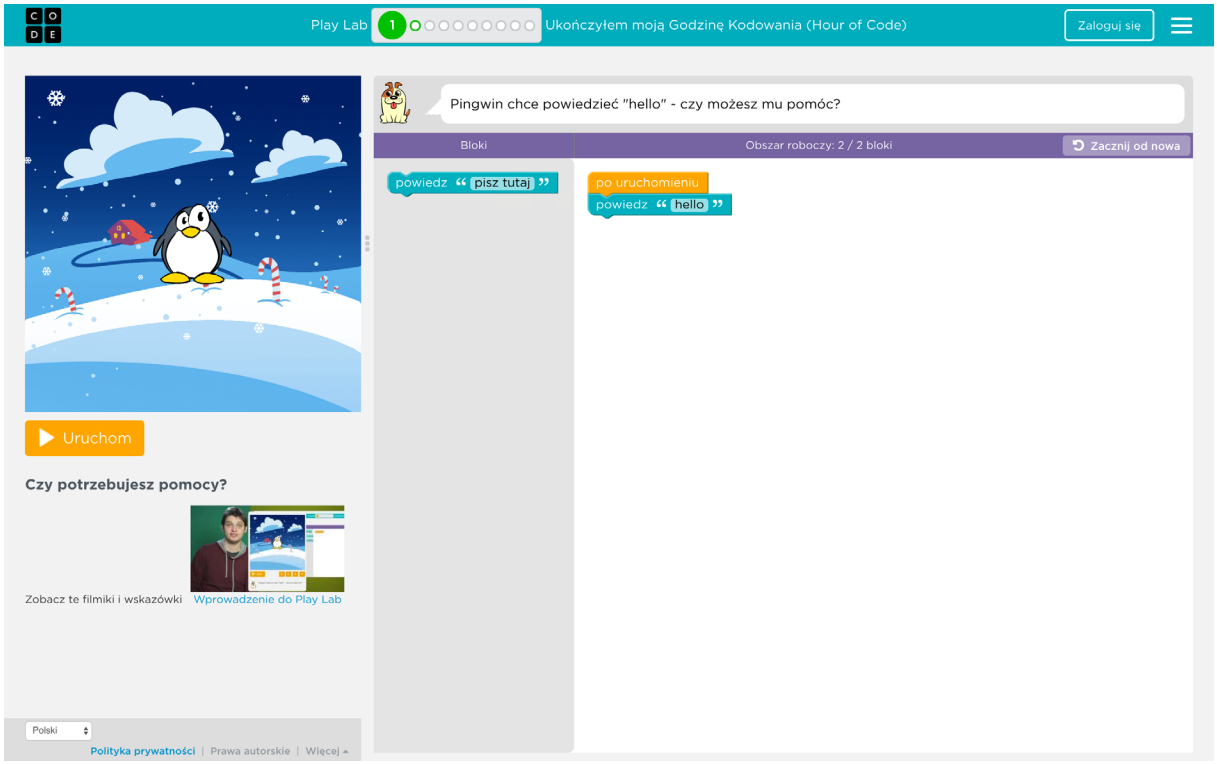


Użytkownik Studio.code.org uczy się uniwersalnego języka programowania.

Im bardziej skomplikowane konstrukcje z bloków, tym bardziej złożony będzie kod, który kryje się pod spodem.

Pingwin chce powiedzieć: „Hello” – czy możesz mu pomóc?

To zadanie (dostępne pod adresem: https://studio.code.org/s/playlab/stage/1/puzzle/1) nadaje się na podsumowanie zajęć przeprowadzonych z wykorzystaniem powyższego przykładu.



Przykład innej lekcji ze strony Studio.code.org wraz z uruchomionym skryptem.

Zadanie dla nauczyciela stażysty

Zadanie 1

Na podstawie wskazówek dotyczących projektowania sytuacji edukacyjnej zawartych w Zeszycie 3 i zamieszczonych w programie pod podanym powyżej adresem instrukcji opracuj plan sytuacji edukacyjnej z uwzględnieniem następujących elementów:

- cel cząstkowy, określający, co należy uzyskać w procesie pracy w danej sytuacji;
- treść (zadanie);
- środki dydaktyczne i metody pracy;
- czas;
- miejsce pracy;
- sposoby sprawdzania wyników.Zadanie 2

Zadanie 2

Na podstawie wskazówek metodycznych zawartych w Zeszycie 1 napisz konspekt lekcji uwzględniający wybrane narzędzia spośród opisanych powyżej.

Zadanie 3

Zapoznaj się z sylabusem kursów 2, 3 i 4 dostępnych na stronie Studio.code.org i odszukaj umiejętności wymienione w podstawie programowej z informatyki dla szkoły podstawowej. Zastanów się, które z zadań można wykorzystać do powtórki materiału i oceny rezultatów.

Podsumowanie

Code.org jest rozwiązaniem popularnym i bezpłatnym, dostępnym na stronie Studio.code.org. Programowanie w tym narzędziu stanowi doskonałe wprowadzenie do korzystania z bardziej zaawansowanych języków. Program umożliwia szybkie projektowanie gier, animacji, rysunków, komponowania muzyki, co ułatwia nauczycielom konstruowanie scenariuszy lekcji z atrakcyjnymi celami i motywującymi, możliwymi do osiągnięcia podczas jednych zajęć rezultatami. Podobnie jak Scratch, Code.org wciąga do zabawy uczniów niezależnie od ich poziomu umiejętności, co ułatwia pracę ze zróżnicowaną klasą. Uczniowie zdolni mogą rozwiązywać trudniejsze problemy lub wspierać innych uczniów.

Strona programu zawiera wiele wskazówek i gotowych projektów, które można przeanalizować lub zmodyfikować. Naszym zdaniem oferuje najlepiej przygotowane kursy do samokształcenia (w zakresie podstaw programowania). Rozwiązanie polecamy zarówno uczniom, jak i nauczycielom, którzy chcą podnieść swoje umiejętności. Kursy i materiały dostępne na Studio.code.org obejmują pełny zakres wymagań podstawy programowej nauki programowania w kształceniu ogólnym.

Tynker.com

Dla uczniów znających język angielski interesującym portalem do nauki programowania może być Tynker.com. Podobnie jak Scratch czy Code.org, oferuje on narzędzia do nauki wizualnego programowania, ciekawie zaprojektowane kursy i samouczki oraz konto, które zapisuje rezultaty i pozwala na współpracę nauczyciela z rodzicami. Opiekunowie mogą śledzić postępy uczniów, nauczyciele proponować alternatywne ścieżki kształcenia. Tynker wspiera naukę programowania nie tylko wizualnego (np. JavaScript, Python) oraz kształcenie umiejętności STEM. Pozwala m.in. programować roboty LEGO WeDo, a także drony. Jest to rozwiązanie płatne, ale zakres materiału, sposób opracowania i możliwości, jakie daje uczniom, zdają się uzasadniać poniesione koszty.

Strona główna portalu Tynker.com

Kursy i materiały dostępne na Tynker.com obejmują pełny zakres wymagań podstawy programowej nauki programowania w kształceniu ogólnym. Podobnie jak w wypadku Code.org, kursy zaproponowane i dostępne na stronie przez autorów systemów stanowią gotowy program zajęć z zakresu programowania.

BeCreo

BeCreo to modułowy zestaw do nauki podstaw programowania, elektroniki, mechatroniki i elementów robotyki. Gra zawiera pełny kurs z wyzwaniami o rosnącym stopniu trudności oraz kilkunastoma, podłączanymi jak klocki, modułami elektronicznymi gotowymi do użycia. W programie szkoleniowym BeCreo uczeń wykonuje proste zadania konstruktorskie, układa program za pomocą wirtualnych bloczków, sterując platformą programistyczną Genuino 101 i modułami wyświetlaczy, diod, głośników, serwomotorów, silników itp. Odczytuje dane z sensorów i na ich podstawie decyduje o działaniu podłączonych urządzeń.

Zestaw do programowania Be Creo.

Jak to działa?

W BeCreo programujemy za pomocą wizualnych bloczków. Od bloczków sterujących diodami, wyświetlaczem, urządzeniami wykonawczymi, bloczków zbierających dane z czujników, po bloczki z podstawowymi programistycznymi instrukcjami (pętlami, zmiennymi, funkcjami itp.). To wszystko pozwala na realizację wielu – co ważne – praktycznych zadań o bardzo różnym stopniu trudności. Zarówno uczeń czwartej, jak i ósmej klasy szkoły podstawowej znajdzie ciekawe zadania dla siebie. Autorzy zapewniają wyposażenie nauczycieli w obudowę metodyczną i pomoce naukowe.

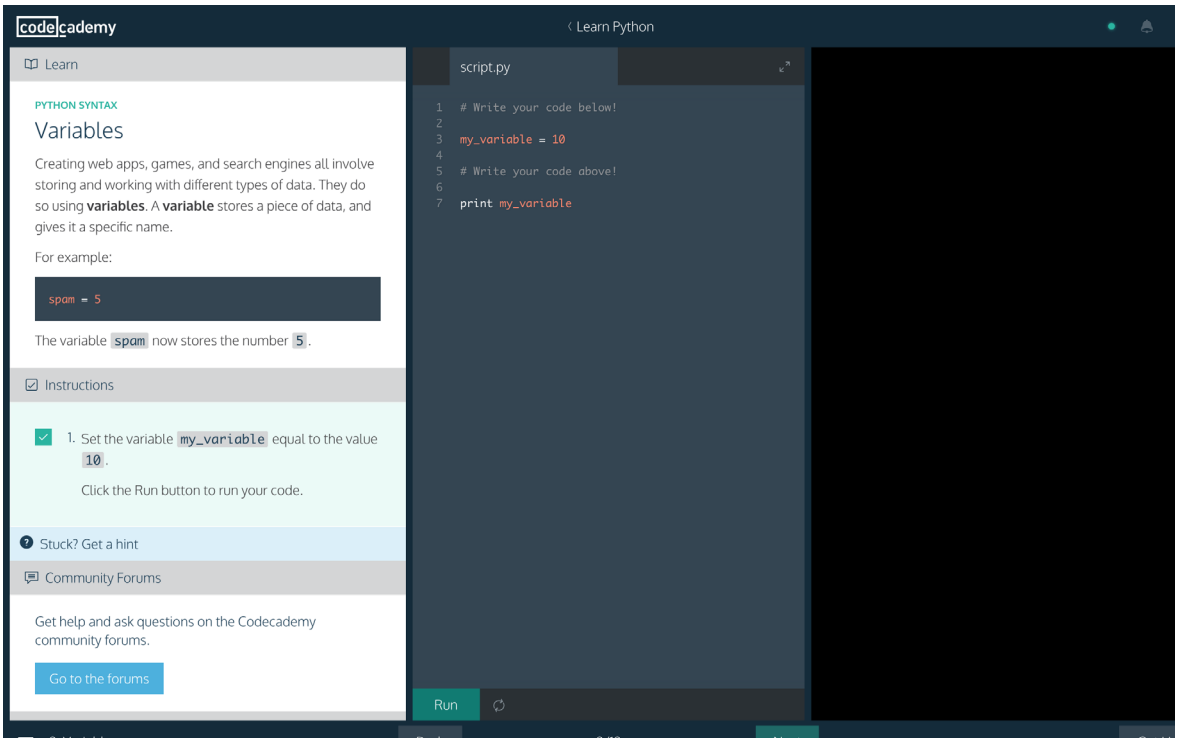
Tryby pracy z BeCreo:

- Samouczek – wprowadzenie do aplikacji BeCreo, dzięki któremu nauczymy się korzystać z wszystkich możliwości oferowanych przez aplikację.
- Kurs – zadania o rosnącej trudności, które stopniowo wprowadzają nas w pełen wachlarz możliwości zestawu BeCreo. W kursie towarzyszy nam instruktor prowadzący nas krok po kroku przez każde zadanie.
- Tryb wyzwań – tematycznie pogrupowane wyzwania, z którymi użytkownik zmierzy się na własną rękę i sam stworzy dla nich rozwiązania.
- Tryb dowolny – freestyle (dostęp do wszystkich funkcjonalności i bloczków w aplikacji).

Praca w trybie dowolnym Be Creo.

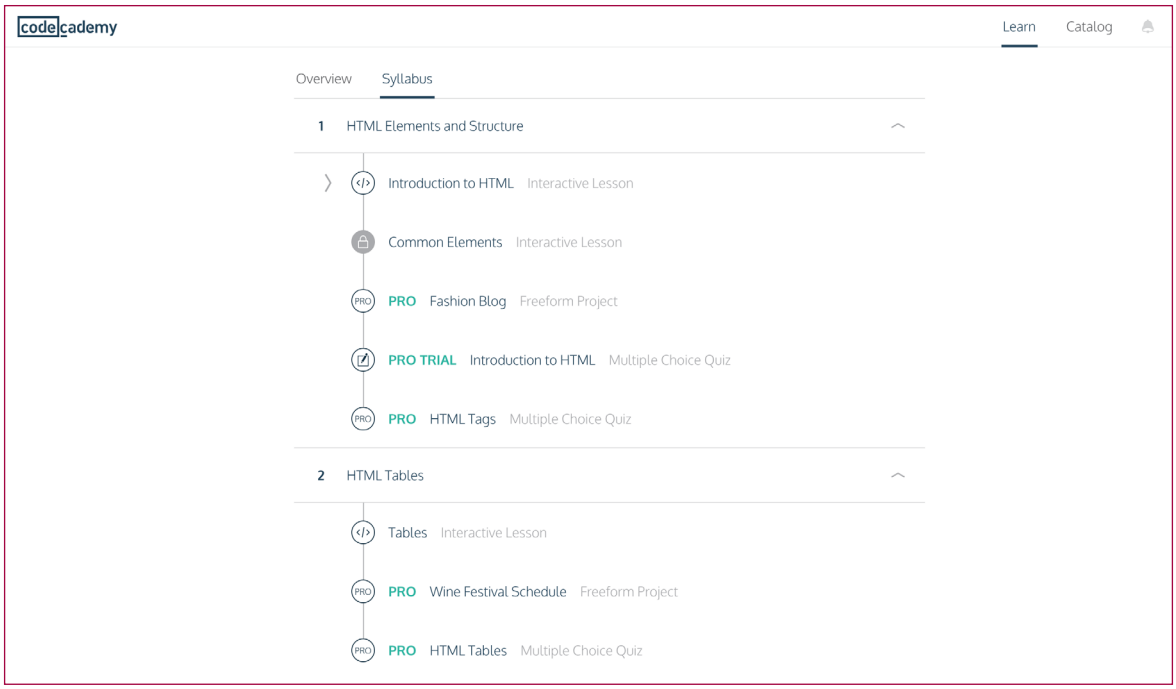
Programowanie w języku Python, HTML, Ruby

Anglojęzyczny serwis Codeacademy.com oferuje metodę analogiczną do Studio.code.org, z tym że pozwala na naukę programowania w językach Python, HTML i Ruby. Jest to serwis płatny, jednak proponowana metoda nauki jest wyjątkowo skuteczna. Nauczyciel i jego uczniowie otrzymują wskazówki krok po kroku, kolejne zadania ułożone są w logicznym porządku. Ich dydaktyczną skuteczność potwierdza szerokie grono klientów (z serwisu korzystają profesjonalne firmy informatyczne do szkolenia swoich pracowników).



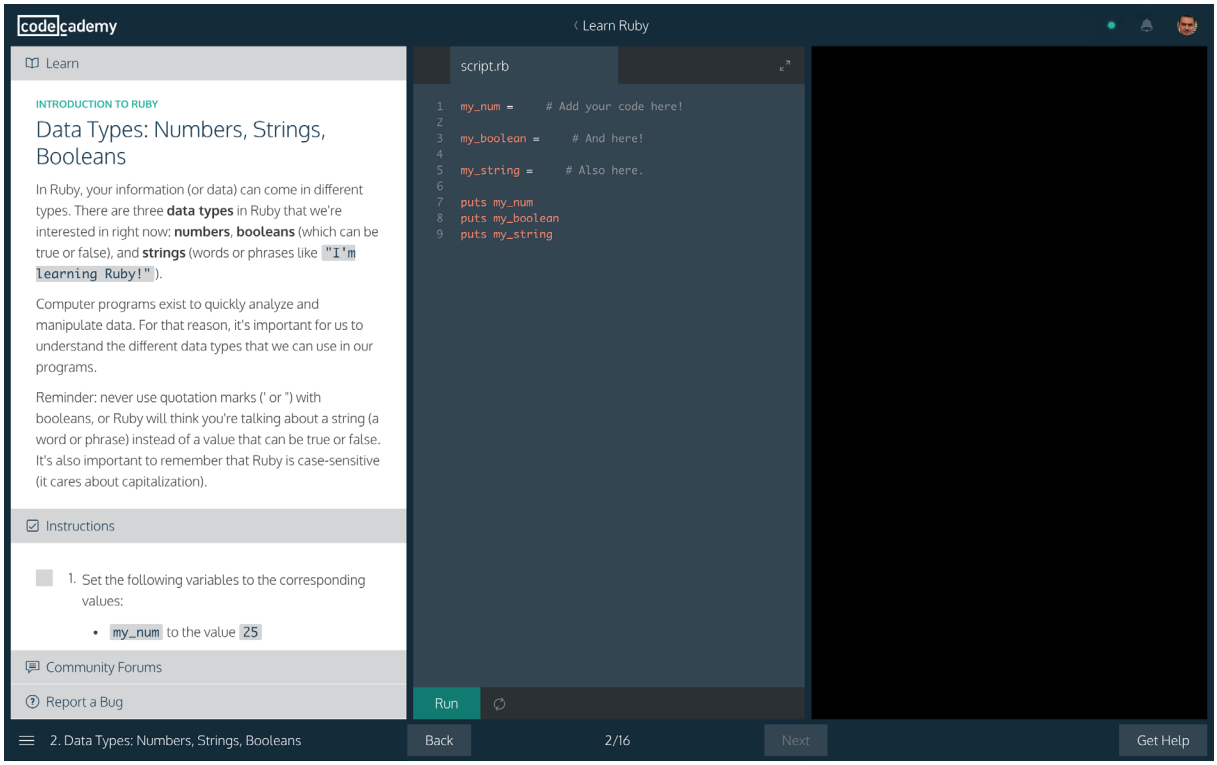
Nauka z Codeacademy.com.

Podobnie jak Code.org platforma Codeacademy.com stawia użytkowników przed coraz trudniejszymi zadaniami. Uczeń (i nauczyciel) jest prowadzony krok po kroku i bardzo szybko osiąga rezultaty w postaci działającej aplikacji.



Lista zagadnień poruszaych w kursie.

System sprawdza kod pisany przez użytkownika, wyświetla informację zwrotną i podpowiada rozwiązania.



Sprawdzanie i podpowiadanie kodu.

Ciekawe alternatywy

Code4startup

Anglojęzyczny płatny portal Code4startup umożliwia naukę programowania przez tworzenie krok po kroku kopii popularnych serwisów i aplikacji, tj. AirBnb, UberEats, Fiverr, ProductHunt. Jest to propozycja wykraczająca poza podstawę programową, ale daje szczególnie wartościową możliwość stworzenia od A do Z dużego projektu w kilku zintegrowanych technologiach. Może to być rozwiązanie pomocne dla klas o profilu informatycznym lub nauki metodą projektu z udziałem uczniów zdolnych. Poszerza perspektywę edukacyjną o elementy związane z nauką przedsiębiorczości, matematyką oraz wszystkimi umiejętnościami miękkimi wpisanymi w podstawę programową (współpraca, nauka w grupie itp.).

Khan Academy

Khan Academy to znany, bezpłatny i popularny portal z filmami instruktażowymi z wielu dziedzin, w tym z informatyki. Prezentuje omówienia podstawowych zagadnień, ale nie daje możliwości przećwiczenia kodowania w wirtualnym środowisku jak np. Codecademy.

Warto zapoznać się także z płatnymi portalami:

- Learn to code at your convenience
- Lynda.com
- UDACITY
- Code School
- SoloLearn
- Pluralsight

Sprawdź, czy potrafisz...

- określić cele i wymagania dotyczące programowania na II etapie edukacyjnym, odwołując się do odpowiednich zapisów w podstawie programowej.
- wymienić i omówić funkcje środków dydaktycznych według koncepcji F. Bereźnickiego i W. Okonia.
- wyjaśnić miejsce programowania w rozwijaniu myślenia komputacyjnego u uczniów.
- porównać funkcje komercyjnych i niekomercyjnych narzędzi do programowania wizualnego.
- wybrać jedno z narzędzi do programowania możliwe do zastosowania na II etapie

  edukacyjnym i omówić jego zalety i wady oraz funkcjonalność.

Dowiedz się więcej

1. Lekcje programowania w polskich szkołach [online, dostęp dn. 15.09.2017].
2. Godzina kodowania – międzynarodowa inicjatywa edukacyjna [online, dostęp dn. 15.09.2017].
3. Darmowy kurs Basic w języku angielskim [online, dostęp dn. 15.09.2017].
4. Materiały do nauki programowania dla nauczycieli i uczniów [online, dostęp dn. 15.09.2017].
5. Ogólnopolska Akademia Programowania [online, dostęp dn. 15.09.2017].
6. Portal informacyjny Pilotaż programowania Ministerstwa Edukacji Narodowej [online, dostęp dn. 15.09.2017].

7. Kurs wprowadzający z informatyki dla uczniów starszych klas szkoły podstawowej oraz dla początkujących nauczycieli w języku angielskim [online, dostęp dn. 15.09.2017].

Bibliografia

Bereźnicki F., (2011), *Zarys dydaktyki szkolnej*, Kraków: Wydawnictwo Impuls.

Duda T., (b.r.), *Funkcje środków dydaktycznych*, [online, dostęp dn. 05.10.2017]

Materiały dydaktyczne udostępnione przez Cybernetic Technologies NETICTECH SA.

Okoń W., (1996), *Wprowadzenie do dydaktyki ogólnej*, Warszawa: Wydawnictwo Akademickie Żak.

Podstawa programowa kształcenia ogólnego z komentarzem. Szkoła podstawowa. Informatyka, (b.r.), [online, dostęp dn. 08.09.2017, pdf. 3,8 MG]

Podstawa programowa z informatyki – szkoła podstawowa, (b.r.), [online, dostęp dn. 20.09.2017, pdf. 206 KB].

Uczymy dzieci programować od najmłodszych lat, (2017), „Nauczyciel i Szkoła" nr 8 (81), s. 12–14.

Spis ilustracji


Fundusze
Europejskie
Wiedza Edukacja Rozwój


ORE
Ośrodek
Rozwoju
Edukacji


Unia Europejska
Europejski Fundusz Społeczny