Centrum Mistrzostwa Informatycznego

Keep scratching!

Scratchowe wyzwania dla zaawansowanych.

Zbiór 24 scenariuszy zajęć na koło programistyczne.

Partner Wiodący Projektu

Politechnika Łódzka

Partnerzy Projektu

AGH

Politechnika Gdańska

Politechnika Warszawska

Politechnika Wrocławska

I love math

Cyfrowy Dialog

Fundusze Europejskie Polska Cyfrowa

Rzeczpospolita Polska

Centrum Projektów Polska Cyfrowa

Unia Europejska Europejski Fundusz Rozwoju Regionalnego

Projekt "Centrum Mistrzostwa Informatycznego" współfinansowany jest ze środków Unii Europejskiej z Europejskiego Funduszu Rozwoju Regionalnego w ramach Programu Operacyjnego Polska Cyfrowa na lata 2014 - 2020

Wstęp

Publikację, do której poznania zachęcamy, kierujemy dla nauczycielek i nauczycieli oraz edukatorów i edukatorek, których podopieczni z klas IV-VIII są już wprowadzeni w podstawy Scratcha.
Aby swobodnie z niej korzystać, niezbędna jest znajomość funkcjonalności tego środowiska programowania oraz swobodne poruszanie się w takich zagadnieniach jak sekwencje, pętle, warunki czy zmienne.
Zbiór 24 propozycji zajęć podzielony został na 4 rozdziały:

PODRÓŻE
KOSMOS
GRY
TAJEMNICE

Każda część zawiera sześć propozycji, które zapraszają do eksperymentowania i odkrywania możliwości Scratcha. Znajdują się w nich wyzwania o różnym stopniu trudności. Mogą być realizowane zgodnie z zaproponowaną kolejnością lub wybiórczo, w zależności od potrzeb oraz możliwości uczestniczek i uczestników zajęć pozalekcyjnych.
W publikacji nie ma gotowych rozwiązań oraz instrukcji „krok po kroku". Można natomiast znaleźć w niej linki do źródłowych projektów oraz wskazówki dla prowadzących zajęcia, pozwalające zwrócić uwagę na te aspekty wyzwań, które mogą wymagać dodatkowego wsparcia w czasie pracy z grupą.
Całość ma charakter otwartego repozytorium możliwości, które daje zabawa w programowanie w Scratchu. Czas do realizacji wyzwań zależy od zaangażowania, umiejętności oraz chęci rozwijania poszczególnych projektów.

Keep scratching!

Autorki

Grażyna Kędzia

Nauczycielka matematyki i informatyki w szkole podstawowej, trenerka podstaw programowania i nowych technologii w Stowarzyszeniu Cyfrowy Dialog. Egzaminatorka OKE egzaminu 8-klasisty z matematyki. Uczestniczka wielu ogólnopolskich projektów edukacyjnych – Centrum Mistrzostwa Informatycznego, Szkoła dla Innowatora. Liderka i trenerka w programach Mistrzowie Kodowania, Szkolni Liderzy Edukacji Zdalnej, Zaprogramuj Przyszłość, Klub Młodego Programisty, Informatyka bez Granic. Autorka wielu innowacji pedagogicznych w zakresie programowania i kodowania. W codziennej pracy wykorzystuje nowoczesne technologie, a wolnych chwilach wędruje po górach i szyje emotkowe pożeracze smutków.

Magdalena Wachulec

nauczycielka informatyki w liceum, egzaminatorka maturalna z informatyki, egzaminatorka ECDL, członkini PTI, trenerka, mentorka i moderatorka w projektach Cyfrowa Szkoła, Aktywna Edukacja, Eksperci Programowania, Centrum Mistrzostwa Informatycznego.
Autorka kursów e-learningowych Nauka poprzez zabawę, czyli programowanie w Scratchu oraz scenariuszy zajęć pozalekcyjnych w ramach projektu CMI.
W ramach projektu CMI prowadziła warsztaty i zajęcia dla nauczycieli z algorytmiki i programowania w Scratchu, tworzyła i weryfikowała zadania konkursowe na potrzeby Ligi Algorytmicznej i Zawodów Algorytmicznych CMI.
Jest współautorką zbioru zadań z informatyki exeBook oraz materiałów do nauki algorytmiki i programowania w C++ (projekt Mermidon).
Jako członkini PTI wielokrotnie zajmowała się organizacją konkursów informatycznych dla dzieci i młodzieży, tworzeniem zadań i ocenianiem prac konkursowych.
Hobby: czytanie książek, gotowanie, łamigłówki logiczne, opera i taniec towarzyski.

Centrum
Mistrzostwa
Informatycznego

Publikacja powstała w ramach Centrum Mistrzostwa Informatycznego.

Parę słów o projekcie Centrum Mistrzostwa Informatycznego

Celem projektu jest podniesienie kompetencji kadry dydaktycznej, tj. osób prowadzących zajęcia pozalekcyjne rozwijające zainteresowania informatyczne, a także aktywizacji młodzieży uzdolnionej informatycznie, pobudzania kreatywności oraz promowania współpracy zespołowej w ramach kół informatycznych.
Projekt CMI stanowi kompleksową koncepcję wzmocnienia polskiej edukacji informatycznej ukierunkowanej na kształcenia uzdolnionych uczniów przy zaangażowaniu najlepszych uczelni technicznych w kraju. Dzięki realizacji wskazanych celów projekt wzmocni u uczniów chęć rozwoju zainteresowań z zakresu algorytmiki i programowania, posłuży także upowszechnieniu idei konkursów informatycznych oraz wyłoni zespoły zdolne do podjęcia rywalizacji w zawodach informatycznych na poziomie krajowym oraz ogólnoświatowym. CMI przyczyni się do podniesienia kompetencji dydaktycznych 1500 osób prowadzących koła informatyczne oraz 12000 uczniów.
Projekt skierowany jest do nauczycieli szkół podstawowych, ponadpodstawowych, nauczycieli akademickich oraz innych osób dorosłych wykazujących predyspozycje do pracy oraz zainteresowania pracą z wybitnie uzdolnioną młodzieżą - uczniów z klas IV-VI szkół podstawowych, uczniów klas VII-VIII szkół podstawowych, uczniów szkół średnich w tym liceów ogólnokształcących i techników oraz szkół branżowych I i II stopnia.
Centrum Mistrzostwa Informatycznego zapewnia uczestnictwo w prestiżowym projekcie o zasięgu ogólnopolskim, w tym możliwość stałego kontaktu ze społecznością renomowanych uczelni technicznych, a także kooperację środowisk dydaktycznych na polu uczelni wyższych i nauczycieli szkolnych.

www.cmi.edu.pl

Spis treści

TEMAT 1

Pamiątki

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- umie korzystać z komunikatów w celu przekazania sterowania do innego duszka;
- umie korzystać z czujników (kliknięcie myszą, dotknięcie innego duszka).

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafiką na wolnej licencji np. pixabay.com.

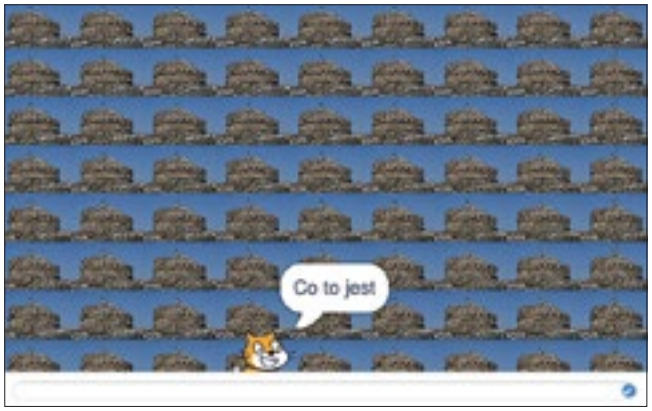
Przebieg zajęć

Wprowadzenie

Podróże po kraju i świecie to czas, w którym poznajemy ciekawe miejsca i ludzi, robimy zdjęcia, wysyłamy kartki do znajomych, kupujemy prezenty, gromadzimy pamiątki i kolekcjonujemy wspomnienia. Spróbujmy uporządkować nasze wrażenia z podróży.

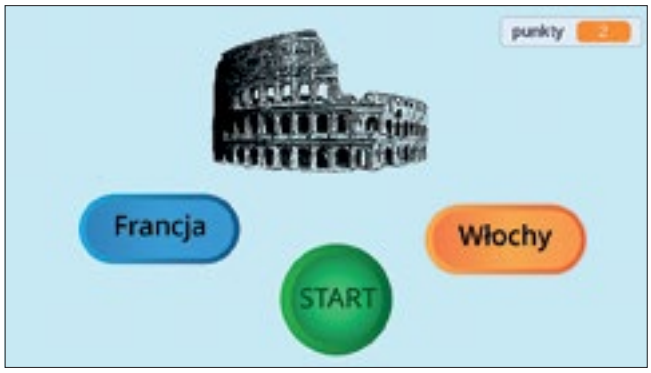
Wyzwanie 1

Napisz program, który będzie wyświetlał ciekawe zdjęcia z wycieczki lub wakacji w formie pokazu. Utrudnieniem dla użytkownika/użytkowniczki powinno być to, że zdjęcia na początku są niewyraźne (np. zastosowano efekt *mozaika* do tła) i dopiero stopniowo stają się coraz lepiej rozpoznawalne. Przygotuj program w formie gry, w której użytkownik/użytkowniczka musi rozpoznać, co widzi na zdjęciu (np. jakie to miasto, jaki to zabytek). Za każdą prawidłową odpowiedź powinny być naliczane punkty. Dodatkowo można wprowadzić premię za szybką odpowiedź lub ograniczyć czas gry.



Wyzwanie 2

Napisz program (grę), który zasymuluje porządkowanie pamiątek z podróży. Należy wybrać dowolną kategorię (np. kraj, rodzaj pamiątki, język, itp.). Duszki-pamiątki powinny pojawiać się na ekranie losowo, zaś użytkownik/użytkowniczka musi je przyporządkować do właściwej kategorii (przeciągnąć myszką w odpowiednie miejsce). Na rysunku pokazano przykład zadania, w którym trzeba rozpoznać, czy dany zabytek mieści się we Włoszech, czy we Francji.



Wyzwanie 3

Napisz swój własny „edytor graficzny", który pozwoli Ci narysować pamiątkową kartkę z wakacji. Program powinien umożliwiać czyszczenie ekranu oraz rysowanie za pomocą myszki. Użytkownik/użytkowniczka może wybierać kolor oraz grubość pisaka.

Opcja dla zawansowanych: wprowadzenie kilku duszków-stempli (np. domy, drzewa), które (po kliknięciu) mogą być dodane do rysunku.

Podsumowanie (wskazówki dla nauczyciela)

Niniejsze wyzwania można wykorzystać do zapoznania uczennic i uczniów z podstawowymi instrukcjami Scratcha. Przykładowe gry są bowiem łatwe w implementacji i nadają się nawet dla bardzo początkujących. Warto zatem szczególnie zwrócić uwagę na estetykę wykonania pracy (dobór zdjęć i duszków) oraz wartość edukacyjną samego programu (można np. zaproponować zaprojektowanie gry dla młodszych koleżanek i kolegów pt. *Zgadnij, jakie to miasto (obraz, zabytek)*, czy *Dopasuj obraz/słowo do wybranej kategorii*. Uczniowie bardziej zaawansowani mogą rozbudować gry, dodając ograniczenia czasowe, punkty karne za błędne odpowiedzi oraz kolejne poziomy trudności.

Stworzenie własnego edytora graficznego pokazuje, jak w prosty sposób wykorzystać reakcje na kliknięcie myszą do „pobierania koloru" i tworzenia rysunku za pomocą myszki.

TEMAT 2

Poliglota

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- potrafi stosować proste operacje na tekstach, w szczególności wypisywać tekst litera-po-literze;
- umie korzystać z list – potrafi dodawać elementy do listy oraz przeglądać listę sekwencyjnie;
- umie korzystać z rozszerzenia Tekst na Mowę oraz Tłumacz.

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafiką na wolnej licencji np. pixabay.com.

Przebieg zajęć

Wprowadzenie

Czy warto uczyć się języków obcych? Oczywiście, że tak. Poznając język i kulturę innych narodów, stajemy się bogatsi wewnętrznie, zyskujemy nowych przyjaciół oraz mamy możliwość podróżowania i zwiedzania ciekawych miejsc w różnych stronach świata. *Podróże kształcą*, dlatego też wybierzmy się w językową podróż po Europie.

Wyzwanie 1

Napisz program, który pomoże w nauce podstawowych słówek i zwrotów w dowolnym języku obcym. Opracuj listę najpotrzebniejszych słówek, następnie napisz skrypt, dzięki któremu zostaną one poprawnie wypowiedziane (użyj rozszerzenia *Tekst na Mowę*).

Wyzwanie 2

Wykorzystaj listy obcych słówek stworzone w poprzednim wyzwaniu. Napisz skrypt, który będzie wyświetlał słowa po polsku i pytał o ich odpowiedniki w wybranym języku.
Program powinien korygować błędy (podawać odpowiedź poprawną) oraz zliczać wszystkie poprawne odpowiedzi.

Wyzwanie 3

Aby lepiej zapamiętać pisownię jakiegoś słowa, literujemy je. Napisz program, który będzie literował słowa w języku polsku i angielskim. Literowanie powinno odbywać się na głos (za pomocą modułu *Tekst na Mowę*). Dodatkowo wypowiadane litery powinny pokazywać się na ekranie (można wykorzystać duszki-litery).

Wyzwanie 4

Napisz program, będzie pełnił funkcję tłumacza z polskiego na angielski i z angielskiego na polski (możesz także wprowadzić inne języki). Uzupełnij listy słówek w każdym języku. W przypadku, gdy słownik-lista nie zawiera określonego słowa, program powinien informować o tym użytkownika (np. komunikatem *Nie rozumiem*).
Dodatkowo możesz wprowadzić opcję *Egzamin*, w której program będzie zadawał pytania (np. po polsku), a użytkownik tłumaczył wyświetlone słowa (na angielski). Pytania powinny być wybierane całego słownika losowo, użytkownik będzie mógł wskazać ich ilość. Na koniec należy podać procent poprawnie udzielonych odpowiedzi.

Podsumowanie (wskazówki dla nauczyciela)

Wszystkie wyzwania dotyczą języków obcych i mogą być opracowane jako projekt grupowy. Zadania, które możemy przydzielić uczniom/uczennicom to:

- opracowanie interfejsu programu (rozplanowanie przycisków np. ćwicz/testuj/literuj/egzamin, przygotowanie duszków-flag oraz duszka-alfabetu, dla którego każdy z kostiumów jest kolejną literą);
- stworzenie skryptów przeglądania listy słówek, wyszukiwania elementu (słówka) na liście, wypisywania tekstu litera-po-literze;
- wykorzystanie modułu Tekst na Mowę do wypowiadania słów w różnych językach oraz modułu Tłumacz do tworzenia list słówek automatycznie np. po hiszpańsku, włosku, francusku, niemiecku (choć oczywiście lepiej skonsultować te tłumaczenia z osobami znającymi dany język);
- stworzenie skryptów sprawdzających poprawność odpowiedzi i liczących punkty oraz skryptów losujących pytania w taki sposób, by pytania się nie powtarzały.

TEMAT 3

Zakupy

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- umie korzystać z list – potrafi dodawać elementy do listy oraz przeglądać listę sekwencyjnie;
- zna operację dzielenia całkowitego, obliczania reszty z dzielenia oraz zaokrąglania do całkowitych (sufit, podłoga);
- poznaje algorytm zachłanny wydawania reszty.

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafiką na wolnej licencji np. pixabay.com.

Przebieg zajęć

Wprowadzenie

Podróże po kraju i świecie to nie tylko czas, w którym poznajemy ciekawe miejsca i ludzi, ale także okazja, by posmakować nowych potraw, zrobić setki zdjęć, a na koniec przywieźć oryginalne pamiątki i upominki dla znajomych. Będąc za granicą, musimy dodatkowo orientować się, jaka waluta obowiązuje w danym kraju oraz umieć przeliczyć ją na złotówki. Spróbujmy rozpocząć podróże ze Scratchem.

Wyzwanie 1

Napisz program, który zasymuluje wypłatę z bankomatu. Po podaniu kwoty do wypłaty skrypt powinien obliczyć i wyświetlić na ekranie, ile banknotów każdego rodzaju zostanie wydanych.

Bankomat zawsze wydaje minimalną liczbę banknotów, czyli stara się zaczynać od jak największych nominałów (np. wypłata 380 zł wymaga jedynie 6 banknotów: 380zł = 3 x 100zł + 1 x 50zł + 1 x 20zł + 1 x 10zł). Dodatkowo program może przeliczać wypłacaną kwotę na inną walutę (np. dolary, euro itp.).

Wyzwanie 2

Napisz program, który zasymuluje zamawianie posiłku w restauracji. Potrawy powinny być wybierane z obrazkowego menu (kliknięcie na potrawę oznacza jej zamówienie). Po złożeniu zamówienia kelner przedstawia rachunek.

Przykładowo: 3 ciastka po 10 zł, 2 kawy po 15 zł, 1 lemoniada za 12 zł, razem: 72 zł.

Wyzwanie 3

Napisz program, który zasymuluje zakupy w sklepie z pamiątkami. Kupujący dysponuje na początku pewną kwotą i nie może jej przekroczyć. Dodawanie do koszyka odbywa się poprzez kliknięcie wybranego przedmiotu. Program powinien blokować zakupy ponad limit, np. podając informację, że brakuje pieniędzy. Na koniec należy wystawić szczegółowy rachunek oraz obliczyć resztę do wydania.

Wyzwanie 4

Zastanów się, jak należałoby zmodyfikować poprzedni skrypt, gdyby oprócz ceny towaru liczyła się też jego waga (bo np. wracamy samolotem i nie możemy zbytnio obciążać walizki) lub gdyby trzeba było uwzględnić „priorytet zakupu" (najpierw znajdujemy prezent dla mamy i taty, a dopiero później drobne upominki dla koleżanek i kolegów). Zapoznaj się z algorytmem plecakowym (*link_1*).

Podsumowanie (wskazówki dla nauczyciela)

W wyzwaniu 1 uczniowie i uczennice poznają algorytm zachłanny wydawania reszty (*link_2*). Dodatkowo, mogą także przeliczać wprowadzane kwoty na inne waluty (np. euro, dolary, funty na złotówki i odwrotnie). Zadania rachunkowe warto urozmaicać za pomocą ciekawego interfejsu (wybór odpowiednich przycisków akcji, duszków-potraw czy duszków-prezentów). Wyzwanie 2 możemy zmodyfikować i zamiast zamawiania potraw – komponować własną pizzę z uwzględnieniem jej wielkości (mała/średnia/duża) i rodzaju ciasta (cienkie/grube) oraz wybierając ulubione dodatki. Na koniec można obliczyć koszt całej pizzy oraz koszt kawałka dla jednej osoby i graficznie „dzielić" pizzę na wybraną liczbę kawałków.

Wyzwanie 3 można sprowadzić do prostego zadania (po kliknięciu na duszka-prezent zmniejsz kwotę do wydania o cenę danego prezentu), przy czym warto jest wprowadzić dodatkowe założenia – wszystkie dane do rachunku (nazwa towaru, cena, liczba sztuk) powinny być zapisywane na liście. Pozwoli to na wyćwiczenie operacji na listach np. przeglądanie, wyszukiwanie, zamiana. Uczniów zaawansowanych dobrze będzie zainteresować również algorytmem plecakowym (*link_1*).

Linki zawarte w opisie

link_1 *link_2*

Palcem po mapie

LINKI DO PROJEKTU

Wyzwanie 1

Wyzwanie 2

Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- stosuje zmienne, pętle, komunikaty;
- wykorzystuje tryb z przeciąganiem/ bez przeciągania;
- porównuje pozycje duszka;
- zna i stosuje czujnik wykrywania indeksu kostiumu.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- dyskusja kierowana,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z pierwowzorem pomysłu z mapą.
- strona z grafiką na wolnej licencji np. pixabay.com;

Przebieg zajęć

Wprowadzenie

Podróżować można na różne sposoby – blisko i daleko, w kilka chwil i godzin albo przez wiele dni, tygodni, lat. Można też podróżować w czasie i przestrzeni, przemierzać galaktyki, poznawać nowe lądy albo... podróżować palcem po mapie. Tematem wyjściowym poniższych zadań jest właśnie mapa świata i Europy. Efektem pracy będą 3 ćwiczenia geograficzne, które pomogą młodszym uczennicom i uczniom w utrwaleniu wiadomości.

Wyzwanie 1

Napisz program pozwalający ułożyć na szablonie mapy puzzle na podstawie kształtów kontynentów. Każde dopasowanie powinno być sygnalizowane dźwiękiem oraz wyświetleniem nazwy kontynentu.

Wyzwanie 2

Stwórz program, który przedstawia mapę np. świata oraz zawiera legendę pozwalającą powiększać i pomniejszać mapę, włączać nazwy kontynentów, oceanów i pasm górskich. Legenda może być rozwijana i zawierać przykładowe przyciski pokazane na rysunku.

Wyzwanie 3

Stwórz program w postaci quizu, w którym wyzwaniem jest rozpoznanie flag krajów europejskich. Po uruchomieniu programu wyświetlają się flagi narodowe – kolejne kostiumy duszka Flagi. Zadaniem użytkownika/użytkowniczki jest kliknięcie w kontur kraju, którego flaga jest wyświetlona na ekranie. Gdy wskaże niewłaściwe państwo, program pozwala podejmować kolejne próby. Za prawidłowe zaznaczenie odpowiedzi gracz/graczka otrzymuje jeden punkt, a duszek *Flagi* zmienia kostium na kolejny.

Podsumowanie (wskazówki dla nauczyciela)

W wyzwaniu 1 wykorzystano czujnik ustaw *tryb przeciągania*. Gdy puzzle zostają rozsypane, jest ustawiany na opcję *z przeciąganiem*, a gdy element umieszczono na właściwym miejscu, zmieniamy na *bez przeciągania*. Tryb bez przeciągania działa poprawnie tylko w wersji pełnoekranowej. W przykładowym rozwiązaniu zastosowano algorytm porównywania aktualnej pozycji x i y danego puzzla z docelowym jego położeniem. Jeżeli bieżąca lokalizacja układanego kontynentu jest w odległości +/- 10 od właściwego położenia, program przyjmuje takie dopasowanie jako poprawne. Stosując to rozwiązanie, warto młodszym uczennicom uczniom przybliżyć porównywanie liczb ujemnych np. -160<-150<-140. Opcjonalnie można zaprogramować pomiar czasu układania puzzli lub dodać kolejne poziomy np. podział administracyjny Europy.

W wyzwaniu drugim wszystkie opcje rozwijalnej legendy są jednym z jej kostiumów. Uruchomienie wybranego wariantu legendy polega na kliknięciu w duszka i wykryciu pozycji *y* myszy. Wówczas następuje wysłanie komunikatu do pozostałych duszków z poleceniami do wykonania, czyli włączenie odpowiednich napisów lub skalowanie mapy. Ponowne kliknięcie w przycisk legendy wyłącza napisy. Możliwości legendy można rozszerzyć o resetowanie powiększenia lub pomniejszenia, zaznaczanie mórz, największych rzek itp.

W ostatnim wyzwaniu czasochłonne może być przygotowanie kostiumów z flagami europejskich krajów oraz duszków będących konturami państw. Celem tego ćwiczenia nie jest wykonanie quizu z wykorzystaniem klocka *zapytaj*, lecz udzielanie poprawnych odpowiedzi poprzez wskazanie (kliknięcie) kształtu kraju, którego flaga jest wyświetlana. Każdy duszek kraju sprawdza po kliknięciu, czy numer kostiumu *Flagi* odpowiada przypisanemu krajowi.

TEMAT 5

Memory

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- stosuje zmienne, pętle, listy;
- wykorzystuje klonowanie duszków.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- dyskusja kierowana,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafiką na wolnej licencji np. pixabay.com, freepik.com.

Przebieg zajęć

Wprowadzenie

Memory to prosta gra, która polega na zapamiętywaniu pozycji kart rozłożonych na przykład na stole. Celem gry jest znalezienie wszystkich par takich samych kart. Jest świetną formą rozrywki – pozwala wspólnie spędzić czas, ale może też posłużyć jako gra dydaktyczna ćwicząca spostrzegawczość i pamięć. Naszym celem jest zaprogramowanie ułożenia 8 kart ze znanymi zabytkami oraz stolicami państw.

Wyzwanie 1

Stwórz skrypt animujący rozłożenie talii składającej się np. z 8 kart – po cztery karty w każdym z dwóch rzędów. Dodaj dźwięk tasowania kart. Wykorzystaj klonowanie duszka.

Wyzwanie 2

Zmodyfikuj program z pierwszego wyzwania, dodając 4 różne kostiumy duszka talia np. grafiki znanych europejskich zabytków. Spraw, by każdy klon otrzymał jeden z kostiumów zgodnie z zasadą: każdy kostium musi być użyty dokładnie 2 razy. Napisz skrypty odpowiadające za sprawdzanie, czy kostiumy dwóch klikniętych klonów są takie same. Jeśli tak, obie karty (klony) powinny zniknąć, jeśli nie, przyjmują kostium „rewers". Algorytm powinien działać, aż wszystkie karty zostaną połączone w pary.

Wyzwanie 3

Wykorzystując skrypty z wyzwania 2, napisz program dobierający w pary dwa różne obiekty np. 4 kraje europejskie z ich stolicami. Możesz dołożyć podpowiedź, która wyświetli się np. po naciśnięciu klawisza spacja – wówczas odsłonią się na chwilkę wszystkie karty.

Podsumowanie (wskazówki dla nauczyciela)

Pierwszym krokiem algorytmu jest ustawienie planszy z odwróconymi kartami. Na tym etapie do losowo ustawionych na planszy kart zostają przypisane różne kostiumy – po 2 każdego rodzaju. W tym celu numerujemy poszczególne klony duszka *talia*, tworząc blok generujący losowe liczby:

Aby do dwóch kart przypisać jednakowy kostium, wykonane zostaje zaokrąglenie z dzielenia wcześniej wylosowanej liczby *nr* przez 2. Dzięki takiemu zabiegowi dokładnie dwie karty będą miały taki sam wynik działania i przypisany zostanie im ten sam kostium. Ważna jest tu zmienna *kostium* stworzona tylko dla danego duszka. Zaznaczenie tej opcji pozwala rozróżnić poszczególne klony. W przypadku zaznaczenia opcji dla wszystkich duszków zmienna *kostium* ustawi taki sam kostium dla wszystkich klonów.

W kolejnym kroku następuje identyfikacja zakrytego obrazka. Po kliknięciu na rewers kostium zostaje zmieniony na przypisany danemu klonowi, a wartość zmiennej *karta1* zostaje ustawiona na numer kostiumu tej karty. Podobnie dzieje się z odkrytą drugą kartą. Użyta zmienna *ile_odkrytych* przyjmuje wartość 1 – jeśli odkryta jest jedna karta, oraz wartość 2 – jeśli odkryte są dwie. Można wówczas przejść do sprawdzenia, czy są takie same. Jeśli obie karty mają ten sam kostium, następuje ich usunięcie, jeśli nie, to znów odwrócone zostają rewersem do góry.

W wyzwaniu 3 użyta została dodatkowa zmienna *kostium2*, która przypisuje każdemu klonowi wartość wylosowanej liczby *nr*. Zmienna *kostium2* pozwala na zastosowanie różnego kostiumu z tej samej kategorii (państwo-stolica) w zależności, czy wynikiem reszty z dzielenia przez 2 jest liczba 0 czy 1. Innymi słowy pozwala ustawić dwa różne kostiumy dla pary kart o tej samej wartości zmiennej *kostium1*. Należy pamiętać, że zmienna *kostium1* przyporządkowuje dwóm kartom na tablicy tę samą wartość, by można je razem usunąć, gdy zostaną jednocześnie odsłonięte. Za odsłonięcie nie odpowiada zatem wygląd kostiumu, ale wartość zmiennej *kostium1* przypisanej do klona.

Skrypt można modyfikować, dodając poziomy trudności 3x4, 4x4, czy 4x5 albo łączyć w pary obrazek z nazwą. Dobrym uzupełnieniem gry będzie zaprogramowanie, w jakim czasie wszystkie pary zostały połączone.

TEMAT 6

Czas na podróż

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- Stosuje zmienne i pętle;
- Korzysta z czujników – aktualny rok, miesiąc, dzień itp.;
- Używa klocków wyrażeń do zaprogramowania obliczeń matematycznych.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- dyskusja kierowana,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafiką na wolnej licencji np. pixabay.com, freepik.com;
- kalkulator czasu na świecie;
- kalkulator odległości.

Przebieg zajęć

Wprowadzenie

Poniższe wyzwania skupione są wokół czasu. Najpierw projektujemy cyfrowy wyświetlacz z bieżącą godziną, datą i dniem tygodnia. Następnie przeliczamy czas w różnych miejscach na świecie. Na koniec obliczamy czas podróży wskazanego odcinka trasy przebytego różnymi środkami transportu.

Wyzwanie 1

Zaprojektuj i zaprogramuj cyfrowy wyświetlacz aktualnego czasu, bieżącej daty i dnia tygodnia.

Wyzwanie 2

Napisz program wyświetlający aktualną godzinę i dzień tygodnia w różnych miejscach na świecie np. Warszawa, Londyn, Nowy Jork i Tokio. Sprawdź, jak przeliczyć czas, na stronie dayspedia.com (*link_1*).

Wyzwanie 3

Napisz program obliczający rzeczywistą odległość w linii prostej pomiędzy wskazanymi punktami na mapie. Następnie oblicz czas, w jakim tę drogę pokonają różne środki transportu (samochód poruszający się z prędkością 90 km/h, samolot lecący z prędkością 800 km/h, rower jadący z prędkością 20 km/h oraz pociąg z prędkością 180 km/h).

Linki zawarte w opisie

link_1 *link_2*

Podsumowanie (wskazówki dla nauczyciela)

W dwóch pierwszych wyzwaniach wykorzystano czujnik wykrywający aktualną godzinę, minutę i sekundę oraz bieżącą datę i dzień tygodnia. Skrypty oparte są w dużej mierze na zmianie kostiumów duszków *godziny, minuty, sekundy, dni tygodnia* itp.

W wyzwaniu 2 należy pamiętać, że podczas przeliczania godziny może nastąpić zmiana dnia tygodnia. Jeśli przesuwamy się w kierunku wschodnim, dzień tygodnia w pewnym momencie zmieni się na następny (po przekroczeniu godziny 24:00), jeśli podróżujemy w kierunku zachodnim, na wcześniejszy (po cofnięciu godziny do 00:00).

W wyzwaniu 3 sceną jest mapa Polski. Na początku należy skalibrować odległości. W tym celu ustawia się współrzędne dwóch dowolnych punktów na mapie (najlepiej miast). Wciśnięcie na klawiaturze liter *s* (start) oraz *k* (koniec) powoduje odczyt współrzędnych x oraz y w miejscu, w którym znajduje się kursor myszy. Następnie twierdzenie Pitagorasa pozwala wyznaczyć odległość między tymi punktami na mapie.

Kalibracja polega na przeskalowaniu otrzymanej odległości do rzeczywistej po wpisaniu nazw miast wskazanych na mapie w poprzedniej części. Wartość rzeczywistą można wyznaczyć przez dowolny kalkulator odległości, np. calc.pl (*link_2*).

Należy pamiętać, że odległość obliczona musi być w linii prostej.

Dzieląc rzeczywistą odległość przez odległość punktów na mapie, otrzymujemy skalę, która posłuży do przeliczenia dowolnej odległości na mapie.

Czas obliczany jest dla każdego pojazdu – duszka – i wyświetlany w formacie 00:00.

TEMAT 1

Kosmiczny inżynier

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- potrafi narysować proste wielokąty foremne, wykorzystując narzędzia z rozszerzenia Pióro;
- potrafi stworzyć własny blok bez parametrów.

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop.

Przebieg zajęć

Wprowadzenie

Kosmiczny Inżynier musi zaprojektować nowoczesny panel sterowania z przyciskami w formie elementów w różnych kształtach. Chciałby ułatwić sobie pracę i zautomatyzować czynności, aby projektowanie i konstrukcja przebiegały szybko i sprawnie. Pomóż mu opracować odpowiednie skrypty do rysowania.

Wyzwanie 1

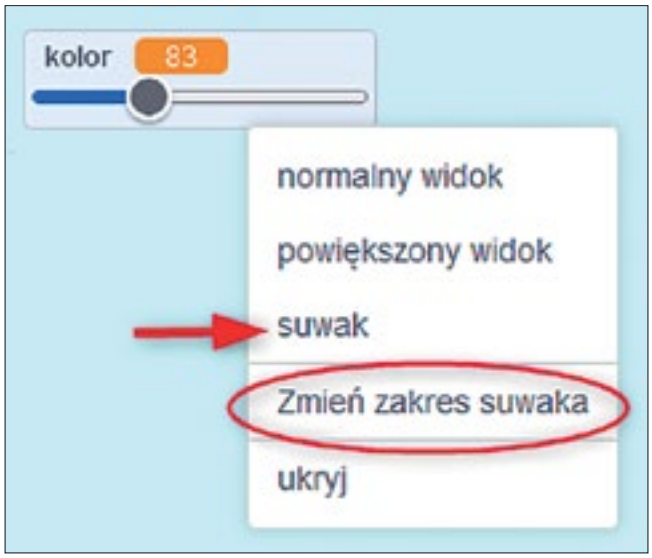
Wybierz duszka *Pencil*, zmień jego rozmiar na 20% oraz ustaw środek ciężkości tak, by ołówek rysował zaostrzonym końcem.

Stwórz projekt, w którym zmodyfikowany ołówek będzie rysował wybrane figury geometryczne (trójkąt, kwadrat, sześciokąt). Rysowanie powinno odbywać się na życzenie użytkownika poprzez wciśnięcie odpowiedniego klawisza klawiatury (**T** – **T**rójkąt, **K** – **K**wadrat, **S** – **S**ześciokąt). Spacja kończy działanie programu.

Czy dostrzegasz pewne analogie? Jak można je wykorzystać?

Wyzwanie 2

Zmodyfikuj projekt tak, by użytkownik mógł decydować o parametrach rysowanych figur (np. ustawiając na suwaku zmiennej kolor pisaka, grubość linii, długość boku wielokąta). Zamiast wciskać klawisze (T, K, S) możesz użyć dodatkowych duszków (liter lub figur geometrycznych), które po kliknięciu lewym przyciskiem myszy przekażą informacje – jaka figura powinna być narysowana.



Wyzwanie 3

Wykorzystaj zależność między liczbą boków wielokąta a kątem obrotu (trójkąt – obrót o 120°, kwadrat – obrót o 90°, n-kąt – obrót o 360/n°) i zaprojektuj uniwersalną procedurę do rysowania dowolnych wielokątów foremnych.

Wyzwanie 4

Wykorzystaj skrypty z poprzedniego wyzwania (możesz je przenieść za pomocą *Plecaka*) i stwórz projekt, w którym *Pencil* – inżynier rysuje dowolną liczbę wybranych wielokątów foremnych. Użytkownik musi mieć możliwość wyboru następujących parametrów:

- kolor i grubość pisaka,
- wielkość elementu – S (small), M (medium), L (large),
- rodzaj elementu – wielokąt foremny o zadanej liczbie boków n ($3<n<12$),
- liczba elementów.

Elementy powinny być narysowane w rzędach poziomych tak, aby nie zachodziły na siebie ani nie wychodziły poza ekran. Próba narysowania większej liczby elementów niż może zmieścić się na ekranie powinna być zatrzymana (ze stosownym komunikatem).

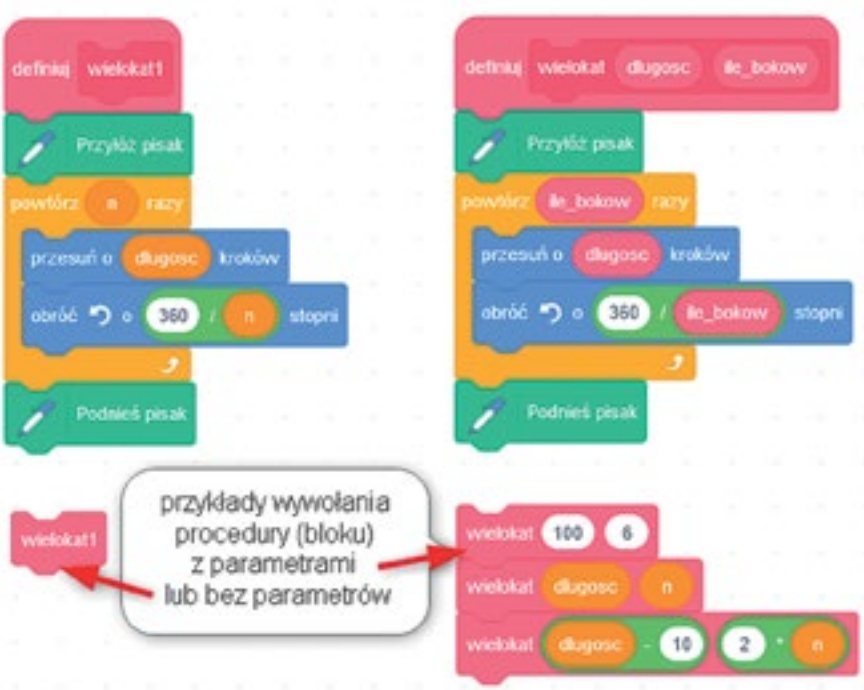
Podsumowanie (wskazówki dla nauczyciela)

W zależności od stopnia zaawansowania grupy można pierwsze wyzwanie potraktować jako rozgrzewkę, która uświadomi uczennicom i uczniom, że do rysowania wielokątów foremnych wystarczy zdefiniować jedną procedurę (własny blok – z parametrami lub bez).

Warto przedyskutować wpływ inicjowania zmiennych na sposób działania programu:

- jak ustalić zakres wartości dla długości boku (czy wielokąt „zmieści się" na rysunku?),
- w którym miejscu ekranu rozpocząć rysowanie,
- jak początkowy kierunek i rodzaj obrotu (lewo/prawo) wpływają na wygląd wielokąta – tzn. czy stoi na podstawie czy jest „do góry nogami".

W grupie początkującej można zdefiniować blok bez parametrów (operując na zmiennych, ustawianych za pomocą suwaka), ale warto wprowadzić pojęcie bloku z parametrami (zwracając uwagę na różnicę między parametrami formalnymi – *bloki różowe* a parametrami aktualnymi, jakimi mogą być stała, zmienna lub wyrażenie). Zwracamy też uwagę na przejrzystość i zwięzłość skryptu wykorzystującego procedury (bloki z parametrem).

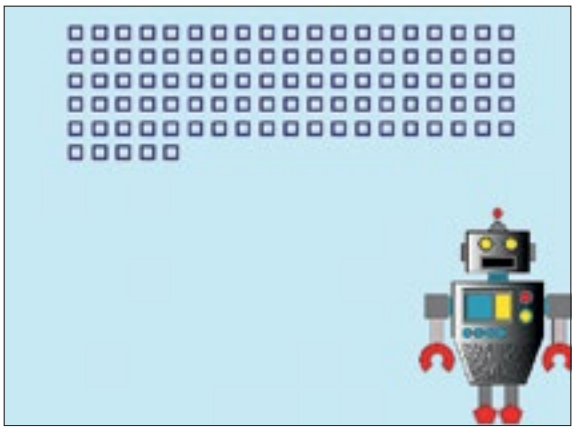


Warto przeprowadzić dyskusję na temat *co i dlaczego powinno być parametrem* oraz *jak najwygodniej wprowadzać dane* (instrukcja *Zapytaj*, wybór wartości na suwaku, czy też, dla zaawansowanych, wczytywanie danych z listy).

Wyzwanie 4 można też potraktować jako projekt grupowy, w którym uczennice i uczniowie muszą samodzielnie zaprojektować interfejs programu oraz sposób eksponowania wyników.

W grupie zaawansowanej należy zwrócić szczególną uwagę na poprawne działanie programu w sytuacjach szczególnych np. gdy użytkownik chce narysować więcej figur niż zmieści się na ekranie lub gdy figury są za duże i nachodzą na siebie lub wychodzą poza ekran.

Wyzwanie 4 – przykładowe rozwiązanie



Notatki do zajęć

TEMAT 2

Kosmiczna podróż

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- potrafi stworzyć własny blok z parametrami;
- stosuje nadawanie i odbieranie komunikatów;
- wie, jak stosować zmienne, instrukcje warunkowe i pętle;
- rysuje wielokąty, rozety przy użyciu rozszerzenia Pióro.

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna,
- praca w grupach.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop.

Przebieg zajęć

Wprowadzenie

Kosmonauta Ludwik planuje kosmiczną podróż po geometrycznych planetach. Na każdej z nich odkryje fantastyczny świat figur. Pomóż mu zaprojektować interfejs projektu oraz aktywności na poszczególnych planetach z wykorzystaniem rozszerzenia *Pióro*.

Wyzwanie 1

Wybierz duszka *Kosmonautę*, dodaj dwie lub trzy planety i zaplanuj kosmiczną podróż po magicznym świecie geometrii.
Stwórz projekt, w którym *Kosmonauta* poleci na planetę, którą wskaże wskaźnik myszy. Gdy Kosmonauta dotknie planety i użytkownik w nią kliknie, ta wyświetli mu swoją nazwę i wyśle wiadomość zapraszającą do zwiedzania.

Wyzwanie 2 – *Planeta 1: Rozety*

Stwórz własnego duszka *Rysik* i zmodyfikuj projekt tak, by wybrana planeta rysowała rozetę, wykorzystując blok z parametrami. Zbuduj własny blok odpowiedzialny za rysowanie wielokąta o parametrach *n* (ilość boków) oraz *długość* (boku). Rozeta powstaje na planie okręgu z *n* wielokątów (*n*) obróconych względem siebie o pewien kąt. Jeśli lubisz kolory, spraw by każdy element rozety był w innym kolorze. Pamiętaj, by pozostałe duszki zostały ukryte.
Spróbuj pobawić się ilością powtórzeń rysowanego wielokąta oraz wartością kąta obrotu.

Wyzwanie 3 – *Planeta 2: Śnieżynki*

Zaprojektuj rysowanie śnieżynek na kolejnej planecie. Podstawowym elementem śnieżynki jest romb o kątach wewnętrznych 60° i 120° oraz losowej długości boku. Wygeneruj 25 takich śnieżynek w losowych miejscach sceny.

Wyzwanie 4 – *Planeta 3: Freestyle*

Zaprojektuj animacje z rysowania figur na kolejnej planecie. Mogą to być gwiazdy, śnieżynki lub fajerwerki. Poeksperymentuj. Zaprezentuj koleżankom i kolegom implementacje swojego pomysłu.

Podsumowanie (wskazówki dla nauczyciela)

Warto sprawdzić, czy uczniowie znają różne sposoby wczytania zmiennej – przy pomocy bloczka *Zapytaj* lub wczytanie z suwaka. Zwróć uwagę, jaki powinien być zakres zmiennej *n* (ilość boków), by powstał wielokąt, a jaką wartość ustawić dla *długości* boku, by figura zmieściła się na ekranie.
Wyzwanie 4 można też potraktować jako projekt grupowy, w którym uczennice i uczniowie muszą samodzielnie zaprojektować nowy blok (procedurę) do rysowania i stworzenia animacji na kolejnej planecie.
W grupie zaawansowanej należy zwrócić szczególną uwagę na poprawne działanie programu w sytuacjach szczególnych np. gdy figury są za duże i nachodzą na siebie lub wychodzą poza ekran.

TEMAT 3

Kosmiczny lot

LINKI DO PROJEKTU

Wyzwanie 1

Wyzwanie 2

Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- dokona symulacji odliczania do startu;
- stworzy skrypt obliczający objętość zbiornika na paliwo;
- zaprogramuje ruch rakiety;
- zastosuje instrukcję warunkową, pętlę *powtarzaj aż*.

JAK PRACUJEMY? (METODY)

- praca w parach,
- burza mózgu,
- praca indywidualna.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona do usuwania tła www.remove.bg.
- strona z grafika na wolnej licencji np. pixabay.com;

Przebieg zajęć

Wprowadzenie

Przestrzeń kosmiczna to fascynujący temat. Wybierając się w kosmiczną podróż, powinniśmy dobrze się do niej przygotować, czyli np. zaplanować taką ilość paliwa, by udało się dotrzeć do celu. Każdy, kto marzy o byciu astronautą, będzie mógł również zaprojektować własny model statku kosmicznego.

Wyzwanie 1

Razem z kolegą lub koleżanką zaprojektuj na kartce model rakiety, którą polecicie na Księżyc. Prototyp rakiety narysuj jako duszka w Scratchu. Możesz również zrobić zdjęcie rysunku i zaimportować (bez tła) jako duszka. Napisz program odliczający do startu rakiety kosmicznej i wyświetlający kolejno odliczane liczby (jako duszek). Możesz wykorzystać rozszerzenie *Zamiana tekstu na mowę*.

Wyzwanie 2

Zmodyfikuj poprzedni projekt. Stwórz odpowiednie kostiumy rakiety i dodaj animację startu oraz lotu. Pamiętaj, że w miarę upływu czasu rakieta oddala się od powierzchni Ziemi, staje się mniejsza, a krajobraz jest inny... Aby dodać realizmu swojej animacji, dołóż lecące meteory lub spadające gwiazdy. Możesz wykorzystać klocki *utwórz klona z siebie* i *kiedy zaczynam jako klon*. Zadbaj o efekty dźwiękowe. Możesz również zaprogramować lądowanie na Księżycu.

Wyzwanie 3

Stwórz program obliczający objętość paliwa, które zmieści się w prostopadłościennym zbiorniku rakiety o wpisanych przez użytkownika/użytkowniczkę wymiarach. Skrypt powinien sprawdzać, czy dana ilość paliwa wystarczy, by dolecieć na Księżyc i wrócić. Jeśli wymiary baku wskażą, że nie zmieści się w nim potrzebne paliwo, program powinien pytać o nowe wymiary zbiornika dopóty, dopóki jego wielkość będzie odpowiednia.

Podsumowanie (wskazówki dla nauczyciela)

Dwa pierwsze wyzwania dotyczą jednego projektu – od narysowania modelu rakiety, po odliczanie do startu i lot na Księżyc. Warto przeznaczyć czas na projektowanie statku kosmicznego i dopracowanie efektów wizualnych lotu, a następnie zaprezentowanie swoich programów rówieśnikom.

W wyzwaniu 1 kolejne liczby podczas odliczania są następnymi kostiumami liczby 10, którą tworzymy, wstawiając kostium 0 do kostiumu 1. W wyzwaniu 2 spadające meteory czy gwiazdy dają efekt poruszającego się tła. Warto dodać ten fragment, by uatrakcyjnić animację lotu.

W wyzwaniu 3 zostały wykorzystane dane dotyczące ilości paliwa niezbędnego do lotu dla rakiety na przykładzie SaturnV (ok. 28 000 m^3).

TEMAT 4

Kosmiczna symetria

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- zna pojęcie symetrii względem osi X i Y oraz początku układu współrzędnych;
- implementuje zależności między współrzędnymi duszków symetrycznych do siebie;
- korzysta z opcji odczytywania pozycji innego duszka;
- stosuje instrukcję warunkową, zmienne, nadawanie i odbieranie komunikatów.

JAK PRACUJEMY? (METODY)

- praca w parach,
- burza mózgu,
- praca indywidualna.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona do rysowania symetrycznych fal weavesilk.com.

Przebieg zajęć

Wprowadzenie

Czy symetria występuje naturalnie w świecie? Wiele obiektów i zjawisk w przyrodzie jest symetrycznych np. od plastrów miodu, płatków stokrotki czy kryształów przez czterolistne koniczyny aż po galaktyki spiralne!
A czym jest symetria? Matematycy określają ją jako operację na obiekcie, po której wykonaniu wygląda on tak samo jak na początku. W celu wprowadzenia uczennic i uczniów w tematykę symetrii można wykorzystać stronę weavesilk.com do rysowania symetrycznych fal w różnych wariantach.

Wyzwanie 1

Napisz program, który zobrazuje symetrię środkową względem początku układu współrzędnych, czyli punktu (0,0). Dodaj 2 takie same duszki (możesz dokonać zmiany koloru jednego z nich, by łatwiej je rozróżniać). Steruj jednym z nich za pomocą myszy lub strzałek, drugi niech „odbije się" zgodnie z symetrią środkową. Dla zobrazowania zasady duszki mogą wyświetlać aktualne współrzędne.

Wyzwanie 2

Napisz program, który będzie tworzył obraz kosmicznej galaktyki, wykorzystując symetrię osiową względem osi X i względem osi Y oraz

symetrię względem punktu. W celu otrzymania śladu użyj stemplowania. Zaprogramuj klawisz zmieniający kostiumy duszka, by efekt graficzny był ciekawszy. Dodaj opcję zmiany rozmiaru duszka.

Wyzwanie 3

Stwórz prostą aplikację do symetrycznego rysowania, wykorzystując rozszerzenie *Pióro*. Zaprojektuj możliwość wyboru, względem której osi symetrii użytkownik/użytkowniczka chce rysować. Pozwól dokonać zmiany symetrii w trakcie rysowania. Dodaj opcję zmiany rozmiaru pisaka. Użyj 2 duszków – jeden będzie rysował tam, gdzie zostanie wciśnięty wskaźnik myszy, drugi będzie odczytywał aktualną pozycję pierwszego duszka i modyfikował ją w zależności od wybranej osi symetrii.

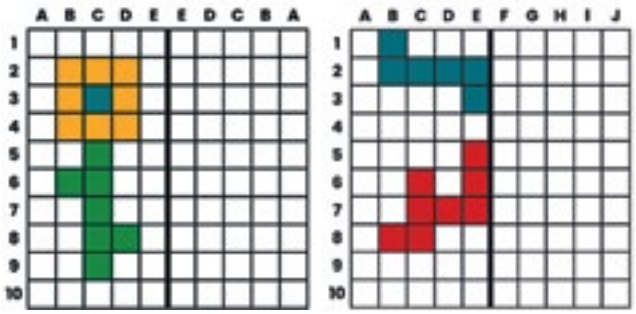
Wyzwanie 4

Napisz program do rysowania symetrii względem przekątnych sceny. Dla pełniejszego obrazowania stwórz skrypt rysujący przekątne w zależności od wybranej opcji symetrii. Zwróć uwagę, że scena w Scratchu nie jest kwadratem. Możesz rozbudować poprzedni program lub stworzyć nowy.

Podsumowanie (wskazówki dla nauczyciela)

Młodszym uczennicom i uczniom należy wytłumaczyć, czym jest symetria osiowa i środkowa. Można zaproponować im zabawy z lusterkiem lub pracę na macie do kodowania (kratownicy), na której będą odwzorowywać obrazki względem osi x oraz osi y. Ciekawą propozycją jest też ćwiczenie rysowania obiema rękami jednocześnie, tak by powstał obrazek osiowosymetryczny.

NIEPODRĘCZNIK 2. Poradnik i scenariusze dla kodujących nauczycielek i nauczycieli; kodowanie bez prądu (link_1)



W wyzwaniu 2 odwołujemy się do współrzędnych wskaźnika myszy, które mnożymy przez -1 w zależności od rodzaju symetrii – albo współrzędną x (symetria względem osi Y), albo współrzędną y (symetria względem osi X), albo obie (symetria względem środka układu współrzędnych (0,0)).

Link zawarty w opisie

link_1

Podczas realizacji wyzwania 3 warto zwrócić uwagę na zmienną przełączającą tryb pisaka (podniesiony czy nie) pozwalającą drugiemu duszkowi sprawdzić, czy został wciśnięty klawisz myszy i czy może rysować symetrię. Zadanie to można zmodyfikować, dodając możliwość wyboru koloru rysowania, rysowania symetrycznej figury lub rozbudować o skrypty z wyzwania 4.

Ostatnie wyzwanie dotyczy symetrii względem przekątnej sceny. W przypadku ekranu kwadratowego symetria względem przekątnej $y = x$ lub $y = -x$ zamienia współrzędne x z y, a w drugim przypadku dodatkowo zmienia znaki obu współrzędnych na przeciwne. Scena w Scratchu jest prostokątem i ma rozmiar 480x360 px. W związku z tym zachodzi równość $4y = 3x$, a stąd $y = 0{,}75x$ oraz $x = 1{,}333y$. Wykorzystamy te zależności do wykonania symetrii względem przekątnej.

Notatki do zajęć

TEMAT 5

Kosmiczne problemy

LINKI DO PROJEKTU

Wyzwanie 1 

Wyzwanie 2 

Wyzwanie 3 

Wyzwanie 4 

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- umie korzystać z list – dodawać i wyszukiwać elementy, wypisywać elementy w kolejności rosnącej/malejącej;
- potrafi zaimportować moduł *Tekst na Mowę* oraz wykorzystać dostępny syntezator mowy;
- potrafi wykorzystać zmienne systemowe dotyczące aktualnego czasu do obliczenia, ile dni lub lat upłynęło od podanej daty (np. daty urodzin).

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strony crazynauka.pl i calcoolator.pl.

Przebieg zajęć

Wprowadzenie

Wyruszając w kosmiczną podróż, stawiamy sobie różne pytania:

- Jak daleko znajduje się cel naszej wyprawy i co o nim wiemy?
- Czy masa i ciężar oznaczają to samo? Jeśli nie, to co wpływa na to, ile waży dany przedmiot? Czy na Ziemi i na Marsie ważylibyśmy tyle samo?
- Skąd wiadomo, ile mamy lat? Czy dzień i rok na innych planetach trwa tyle samo co na Ziemi?

Nasz komputer pokładowy musi sprawnie udzielać wszystkich potrzebnych informacji oraz przeprowadzać dla nas skomplikowane obliczenia.

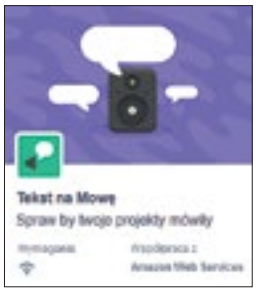
Wyzwanie 1

Napisz program, który wypisze nazwy wszystkich planet Układu Słonecznego oraz poda ich odległość od Słońca. Możesz zacząć od najbliższej lub najdalszej planety. Nazwy planet powinny być wyświetlane przez duszka.

Wyzwanie 2

Wyszukaj w Internecie więcej informacji na temat tych planet. Oprócz danych liczbowych takich jak np.: masa, średnica, przyspieszenie grawitacyjne, okres obrotu wokół osi (dzień), czas obiegu wokół słońca (rok) mogą to być cechy danej planety oraz ciekawostki na jej temat. Zapisz informacje w pliku tekstowym, następnie zaimportuj te dane do listy (lub kilku list).

Korzystając z modułu *Tekst na Mowę*, napisz program, który głosem komputera zaprezentuje wszystkie informacje dotyczące wybranej planety. W przypadku, gdy użytkownik zażąda informacji o planecie spoza naszego Układu Słonecznego, program powinien podać odpowiedni komunikat np. *nie znam takiej planety*.



Wyzwanie 3

Napisz program, który przeliczy wagę dowolnego ziemskiego przedmiotu podaną w kilogramach na ciężar tego przedmiotu na innej (wybranej) planecie.

Wyzwanie 4

Napisz program, który po wczytaniu daty urodzenia obliczy wiek danej osoby. Wiek należy podać w latach, przy czym trzeba pamiętać, że przed urodzinami jesteśmy „o rok młodsi". Przykładowo: osoba urodzona 1.02.2000 w dniu 31.01.2023 ma jeszcze 22 lata, dopiero dzień później skończy 23 lata.

Dodatkowo dla chętnych – obliczenie wieku w dniach oraz sprawdzanie poprawności wczytywanych danych.

Wyzwanie 5*

Wykorzystaj zgromadzone informacje o innych planetach do stworzenia w Scratchu ciekawej prezentacji o naszym Układzie Słonecznym. Zaprojektuj aplikację, która po podaniu daty urodzenia i wagi obliczy wiek (w latach lub dniach) oraz ciężar na wybranej planecie.

Podsumowanie (wskazówki dla nauczyciela)

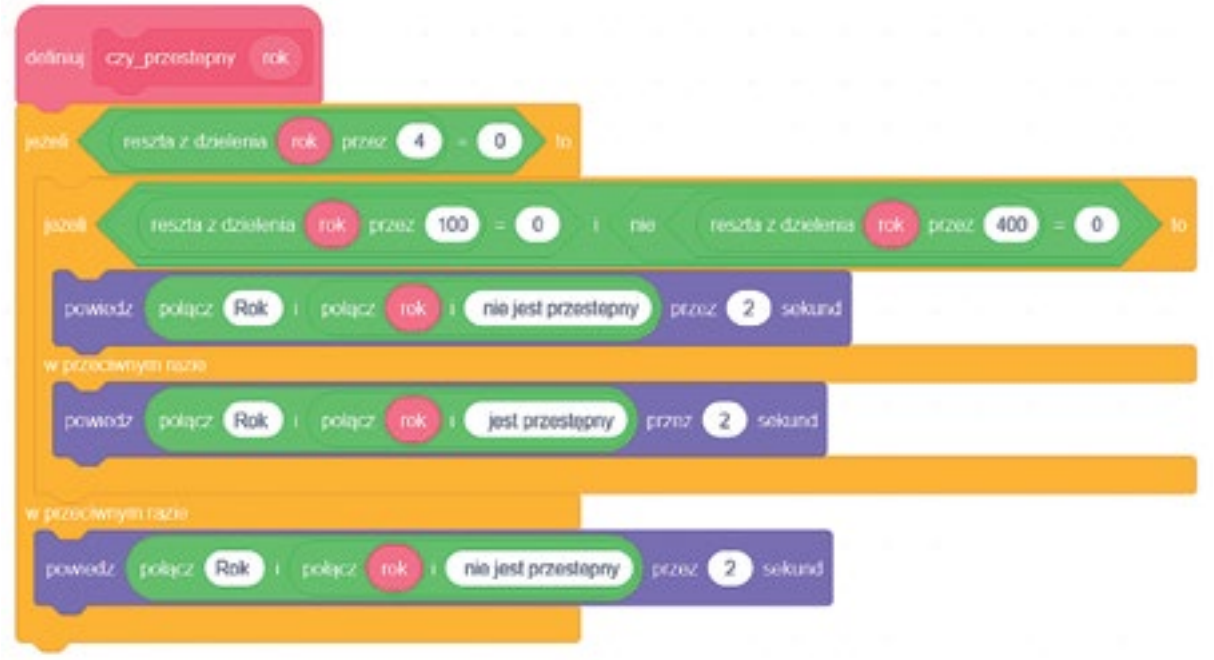
W zależności od stopnia zaawansowania i wieku uczennic i uczniów można: albo poprzestać na początkowych, prostych wyzwaniach, albo zająć się realizacją większego projektu (wyzwanie 5) i przygotować pomoc naukową z zakresu astronomii. Dla młodszych uczennic i uczniów szczególnie atrakcyjne wydaje się wykorzystanie modułu *Tekst na Mowę* (zamiast zwykłego bloczka *Powiedz*). Warto też zwrócić uwagę, że zapisywanie tekstów w liście daje nam szereg korzyści:

- długie teksty są bardziej czytelne i łatwe do modyfikacji;
- program jest krótszy i bardziej przejrzysty;
- informacje można zapisywać w *Notatniku*, następnie importować do wybranej listy.

Starsze uczennice i starszych uczniów możemy zainteresować projektem grupowym, w którym wykorzystają duszki stworzone za pomocą darmowych zdjęć planet. Zastosują przy tym proste obliczenia i animacje ukazujące wielkość planety w porównaniu z Ziemią (proporcjonalna zmiana rozmiaru duszka-planety) oraz przedstawią odległość w jednostkach astronomicznych (1 jednostka astronomiczna to średnia odległość między Ziemią a Słońcem, 1 AU = 149 597 870 700 m). Jako że pracujemy na bardzo dużych liczbach, należy podkreślić, że wszystkie obliczenia są przybliżone, zaś błędy zaokrągleń mogą być niekiedy dosyć duże.

Praca w grupach da ponadto możliwość kooperacji, uczenia się od siebie nawzajem oraz wykorzystania mocnych stron wszystkich uczestniczek i uczestników zajęć. Uczniowie i uczennice mogą zadecydować, które role i zadania najlepiej im pasują (wyszukiwanie i systematyzowanie informacji, projektowanie interfejsu graficznego, tworzenie animacji, tworzenie algorytmów i wykonywanie obliczeń itp.). Obliczenie wieku w dniach może sprawić pewną trudność, dlatego jest to zadanie dla chętnych. Warto zwrócić uwagę na konieczność sprawdzania poprawności danych, aby program był odporny na błędy wynikające z wpisania nieistniejącej daty. Podczas obliczania wieku w dniach można wykorzystać listę zawierającą liczbę dni w poszczególnych miesiącach oraz skrypt sprawdzający, czy dany rok był przestępny.

Tak sprawdzamy, czy rok jest rokiem przestępnym



Notatki do zajęć

TEMAT 6

Kosmiczne piękno

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- potrafi tworzyć własne bloki z parametrem;
- zna i rozumie pojęcie rekurencji;
- potrafi stworzyć proste skrypty rekurencyjne do rysowania fraktali.

JAK PRACUJEMY? (METODY)

- pogadanka/ wykład/ prezentacja,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- dużo świetnych animacji w Scratchu;
- program do rysowania fraktali;
- fraktale w naturze oraz informacje dotyczące fraktali, ciekawostki, przykłady:

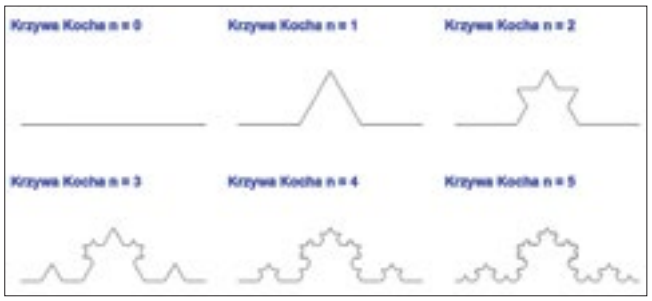
Przebieg zajęć

Wprowadzenie

Gdy patrzymy na otaczający nas świat, zauważamy piękno przyrody oraz niezwykłą różnorodność obiektów. Wpatrując się jednak uważniej, dostrzeżemy, że pod pozornym chaosem i przypadkowością kryje się często pewna regularność. Wystarczy obejrzeć różne zdjęcia (np. brokuły, paprocie, płatki śniegu), by zauważyć samopowtarzalność obiektów, w których część podobna jest do całości. Takie samopodobne obiekty nazwano fraktalami. Badają je matematycy, obserwują biolodzy i geografowie, a informatycy wykorzystują np. do generowania abstrakcyjnych krajobrazów i roślin. Podziwiając piękno fraktali, spróbujmy samodzielnie narysować niektóre z nich.

Wyzwanie 1

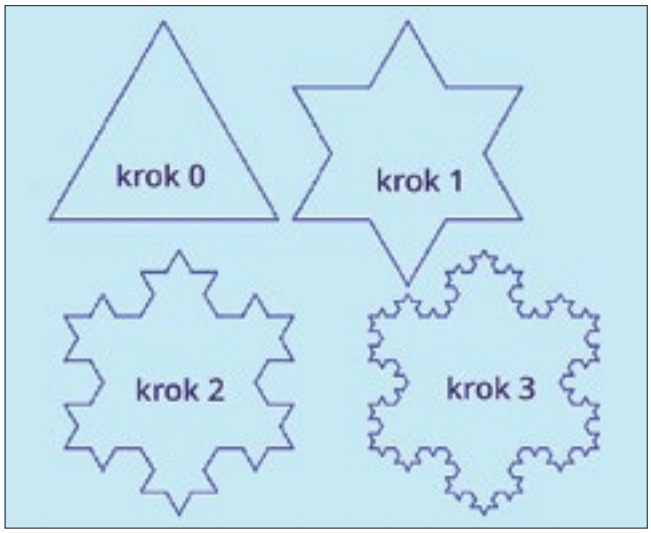
Krzywa Kocha powstaje z odcinka podzielonego na równe 3 części, w którym środkowa zastąpiona zostaje „ząbkiem" o ramionach równych 1/3 odcinka (wraz z usuniętą częścią stworzyłby trójkąt równoboczny). Krok ten jest powtarzany w nieskończoność, dla każdego fragmentu odcinka. Na rysunku przedstawiono kolejne etapy tworzenia krzywej Kocha.



Napisz skrypt, który przedstawi etapy rysowania tej krzywej (od 0 do 5).

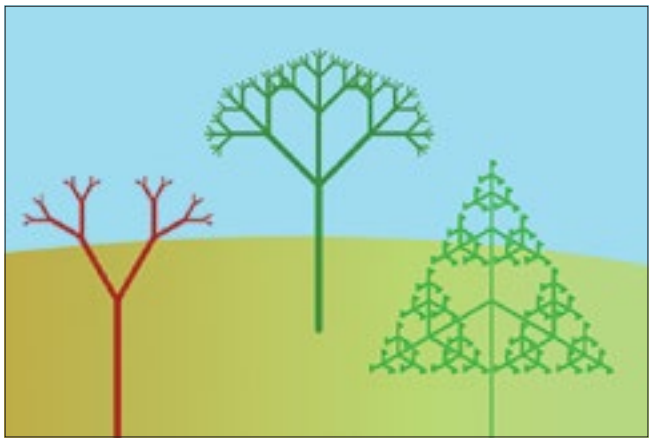
Wyzwanie 2

Płatek Kocha (śnieżynka) składa się z trzech krzywych Kocha. Wykorzystaj poprzedni skrypt, aby narysować płatek Kocha dowolnego stopnia. Zauważ, że wraz ze wzrostem złożoności rysunku czas rysowania mocno się wydłuża.

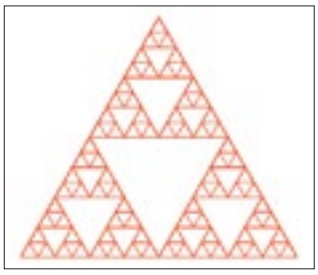


Wyzwanie 3

Na planecie Fraktalli rosną różne gatunki drzew. Napisz skrypty rysujące wybrane drzewa (np. takie jak na obrazku poniżej). Możesz także wymyślić nowe odmiany roślin.



Wyzwanie 4



Mieszkańcy Fraktalli mieszkają w piramidach, które budową przypominają trójkąt równoboczny z wyciętym w środku mniejszym trójkątem. Wierzchołki małego trójkąta znajdują się w połowie boków dużego. Czy potrafisz opisać proces konstrukcji takiej figury? Jest to fraktal znany pod nazwą trójkąta Sierpińskiego. Zaprojektuj domy dla kolejnych pokoleń Fraktalijczyków.

Podsumowanie (wskazówki dla nauczyciela)

Pojęcie rekurencji jest dosyć trudne, dlatego można je przybliżyć za pomocą przykładów z życia (odbijające się obrazy w lustrze, lalki matrioszki, itp.). Dobrze jest skorzystać z dostępnych prezentacji lub filmów dotyczących fraktali oraz pokazać proces tworzenia fraktali krok po kroku za pomocą animacji (bez wchodzenia w zaawansowaną matematykę). Następnie zachęcamy uczennice i uczniów do samodzielnego narysowania na kartkach kolejnych etapów powstawania śnieżynki Kocha czy trójkąta Sierpińskiego.

Mniej zaawansowani uczniowie będą potrzebowali wsparcia nauczyciela lub nauczycielki, aby stworzyć skrypt z wyzwania 1 (krzywa Kocha), natomiast narysowanie śnieżynki nie powinno im już sprawić trudności. Jeżeli poradzą sobie z początkowymi wyzwaniami, można zaproponować im pracę w grupach – tworzenie różnych drzew lub projektowanie własnych fraktali, które następnie zaprezentują pozostałym uczennicom i uczniom.

Notatki do zajęć

TEMAT 1

Złap muchę

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- potrafi stworzyć animację poklatkową z kostiumów duszka;
- potrafi zaprogramować prostą grę zręcznościową;
- dodaje do gry punkty, czas gry;
- stosuje nadawanie komunikatów w Scratchu.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- przykładowe gry edukacyjne i zręcznościowe.
- strona z grafika na wolnej licencji np. pixabay.com;

Przebieg zajęć

Wprowadzenie

Pierwowzorem poniższego zadania jest popularna gra „Whack a mole", która polega na szybkim uderzaniu plastikowymi młoteczkami pojawiających się głów kolorowych bobrów. Gra jest dostępna w wersji planszowej, online'owej oraz edukacyjnej np. na wordwall.com.

Głównym zadaniem jest zaprogramowanie gry zręcznościowej, w której należy złapać muchy w określonym czasie. Mucha pojawia się i znika w dziurze, co utrudnia jej schwytanie. Za każde trafienie muchy packą, zyskujesz jeden punkt. Po upływie czasu program będzie wskazywał zdobytą ilość punktów.

Wyzwanie 1

Stwórz animację duszka *Mucha*, który będzie wychodził z dziury i chował się do niej. Przygotuj odpowiednią ilość kostiumów. Najpierw wybierz duszka, przejdź do sekcji *Kostiumy* i narysuj kształt, który będzie symbolizował dziurę. Zduplikuj ten kostium odpowiednią ilość razy i modyfikuj, usuwając za pomocą gumki coraz mniejsze fragmenty wyglądu duszka. Pracuj w trybie bitmapy. Ułóż skrypt zmiany kostiumów.

Wyzwanie 2

Do poprzedniego projektu dodaj packę na muchy (narysuj lub zaimportuj ilustrację pobraną z sieci). Zaprogramuj packę tak, by zawsze i przesuwała się w stronę wskaźnika myszy, a po wciśnięciu klawisza spacji obracała się, animując trafienie w muchę. Zmodyfikuj skrypt muchy tak, by wykrywała uderzenie i dodawała jeden punkt. Dołącz efekty dźwiękowe.

Wyzwanie 3

Gotowego duszka *Mucha* skopiuj kilka razy i rozmieść na planszy. Zadbaj, by każda kopia rozpoczynała animację wychodzenia i chowania się w dziurze w innym czasie. Zaprogramuj odpowiedni układ duszków na scenie. Dodaj ograniczenie czasowe gry np. na 20 s. Możesz stworzyć duszka wyświetlającego upływający czas. Na koniec gry dodaj zwycięskie tło i wyświetl komunikat z ilością zdobytych punktów.

Podsumowanie (wskazówki dla nauczyciela)

Przygotowując kostiumy *Muchy* do animacji, można pierwszy kostium zduplikować np. 6 razy i na każdym kolejnym dokonywać modyfikacji. Pozostałe kostiumy animujące chowanie się do dziury można skopiować z wcześniej powstałych.

W animacji zmiany kostiumu warto dać krótki odstęp czasu między kolejnymi klatkami np. 0.1 s, daje to wrażenie płynniejszego ruchu i pozytywnie wpływa na efekt całej gry.

Jeśli wystarczy czasu, można wprowadzić modyfikacje w postaci menu startowego, poziomów zaawansowania gry, ilości żyć itp.

TEMAT 2

Gramy w kości

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- umie generować wartości losowe z dowolnego zakresu.

JAK PRACUJEMY? (METODY)

- pogadanka,
- praca indywidualna.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop.

Przebieg zajęć

Wprowadzenie

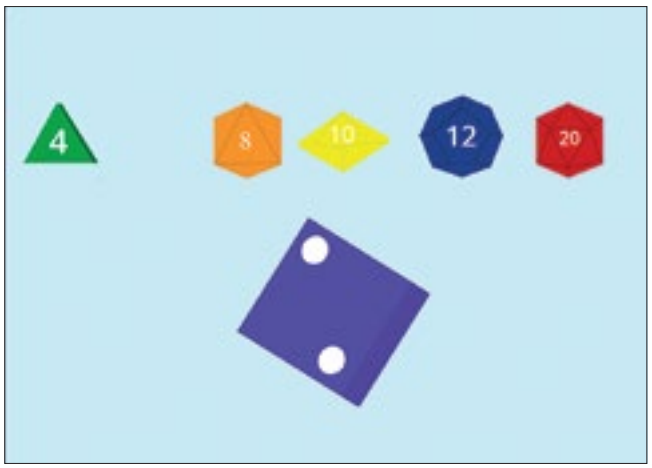
Dawno, dawno temu, gdy nie było jeszcze komputerów, dzieci i dorośli lubili grać w kości bądź planszówki. Obecnie planszówki znów „wracają do gry", a kości (nie tylko te zwykłe sześcienne, ale dwunasto- i dwudziestościenne) bardzo przydają się w popularnych RPG-ach. Spróbujmy wykorzystać Scratcha do symulowania rzutów kostkami.

Wyzwanie 1

Utwórz duszka Kostka, którego 6 kostiumów będzie obrazowało 6 ścianek kostki z liczbą oczek od 1 do 6. Napisz skrypt symulujący ruch kostki. *Kostka* powinna chwilę się obracać lub turlać, wydając odpowiednie dźwięki, a po zatrzymaniu pokazywać losowo wybrany kostium (liczbę wyrzuconych oczek).
Dodatkowo program powinien zliczać, ile razy wystąpiły wartości 1, 2, ..., 6.

Wyzwanie 2

Zaprojektuj inne duszki *Kostki* (w formie wielościanów foremnych). Zmodyfikuj skrypt z wyzwania 1, aby symulować rzuty tymi kostkami.



Wyzwanie 3

Wykorzystaj duszka *Kostka* (z wyzwania 1) do stworzenia prostej gry, w której osoby grające naprzemiennie rzucają kostkami. Wygrywa ten, kto po serii rzutów uzyskał w sumie większą liczbę oczek. Możesz także wymyślić własne zasady gry i dostosować do nich swój skrypt.

Wyzwanie 4

Wykorzystaj duszka *Kostka*, aby zaprogramować wyścigi dwóch duszków. Każdy duszek porusza się o tyle kroków (krok = 10 pikseli), ile wskazuje liczba wyrzuconych na kostce oczek. Wygrywa ten, kto wcześniej dotrze do mety.

Podsumowanie (wskazówki dla nauczyciela)

W wielu programach przydaje się umiejętność generowania liczb losowych oraz losowa (bądź sekwencyjna) zmiana kostiumu. Warto zacząć od prostej symulacji rzutu kostką (ruch, dźwięk), ale starszym, bardziej zaawansowanym uczennicom i uczniom można zaproponować generowanie liczb losowych, zapisywanie ich na liście oraz użycie list (zamiast zmiennych) do zliczania poszczególnych wartości. Wyzwanie 1 może być zatem inspiracją do różnorodnych zabaw z listami.
Uczniowie i uczennice, ćwicząc operacje na listach, mogą wymyślać własne gry, w których wygrana jest określona za pomocą dowolnych kryteriów, np.:

- wygrywa ten, kto uzyskał większą sumę liczb parzystych/nieparzystych;
- wygrywa ten, kto więcej razy wyrzucił szóstkę;
- wygrywa ten, kto więcej razy wyrzucił dwa razy z rzędu te same liczby;
- wygrywa ten, kto „założył się" z komputerem, że osiągnie określony wynik (np. suma oczek w 10 rzutach wyniesie ...) i był bliżej celu.

Generator liczb losowych (z dowolnego zakresu) można wykorzystać także do tworzenia danych testowych dla innych programów, wymagających przetwarzania dużej liczby danych (np. 10 000, czy 200 000).

TEMAT 3

Dot.io

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- Stosuje zmienne, instrukcje warunkowe, komunikaty i pętle;
- Stosuje różne sposoby sterowania obiektem;
- Potrafi wykorzystać klonowanie duszków.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafika na wolnej licencji np. pixabay.com.

Przebieg zajęć

Wprowadzenie

Agar.io to przykład wieloosobowej gry przeglądarkowej stworzonej przez Matheusa Valadaresa. Rozgrywka polega na sterowaniu komórki na kratownicy tak, by dzięki wchłanianiu mniejszych komórek, uzyskała jak największą masę. Nazwa gry pochodzi od agaru, substancji używanej do karmienia kultur bakterii. Gra szybko okazała się sukcesem i została jedną z najbardziej popularnych gier przeglądarkowych i mobilnych 2015 roku (źródło Wikipedia).

Wyzwanie 1

Zaprojektuj ekran startowy i stwórz Menu gry, czyli dodaj przyciski *Graj, Opcje, Info* oraz przycisk *Back*, pozwalający wrócić do Menu. Po wciśnięciu *Graj* powinno załadować się gładkie tło, a wszystkie przyciski powinny zostać ukryte. Przycisk *Opcje* pozwala włączyć i wyłączyć muzykę oraz sterować jej głośnością. Pod przyciskiem *Info* umieść zasady gry oraz informację o jej pierwowzorze. Pamiętaj, że po kliknięciu konkretnego przycisku pozostałe są niewidoczne.

Wyzwanie 2

Do poprzedniego projektu dodaj 3 duszki: *Dot, Pokarm, Krata*.

Duszek *Krata* odpowiada za stworzenie planszy gry, która sterowana jest za pomocą wskaźnika myszy i daje wrażenie poruszania się duszka *Dot*.

Duszek *Pokarm* powstaje poprzez klonowanie siebie. Klony są w różnych kolorach i pojawiają się w losowych miejscach planszy. Po dotknięciu przez duszka *Dot* klon *Pokarm* zostaje zjedzony, czyli usunięty z planszy, co jest sygnalizowane dźwiękiem. Każdy *Pokarm* zjedzony przez duszka *Dot* powoduje doliczenie jednego punktu oraz zwiększenie rozmiaru *Dota*.

Wyzwanie 3

Zmodyfikuj tempo poruszania się *Kraty* w zależności od ilości zjedzonego *Pokarmu*. Im wiekszy *Dot*, tym wolniej się rusza. Wprowadź możliwość zmiany kostiumu w zależności od ilości uzbieranych punktów.

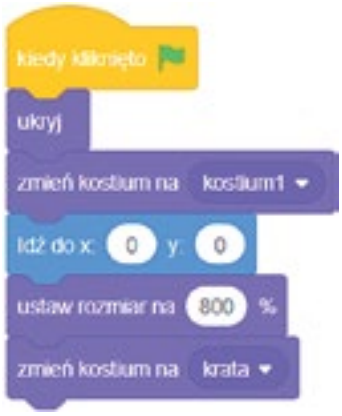
Dodaj i zaprogramuj duszka *Wirus*, który powoduje utratę 2 punktów w przypadku zjedzenia go przez *Dot*.

Wyzwanie 4

W miarę możliwości dodaj zmienną chmurową przechowującą informację o wysokości rekordu punktowego danej rozgrywki. Możesz wprowadzić ograniczenie czasu gry, rozbudować duszka *Wirus* np. o możliwość różnicowania punktacji ujemnej w zależności od swojego wyglądu.

Podsumowanie (wskazówki dla nauczyciela)

W programie wykorzystany jest czujnik odczytywania współrzędnych wskaźnika myszy do sterowania planszą, czyli duszkiem *Krata*. Duszek powinien posiadać 2 kostiumy: pusty (kostium1) i krata oraz mieć ustawiony duży rozmiar np. 800%. Pozwala to na sterownie i przesuwanie kratownicy, co imituje ruch duszka *Dot*.



Po modyfikacjach gry należy uaktualnić treści zawarte w *Info*, np. w postaci kolejnych kostiumów albo wykorzystania rozszerzenia *Zamiana tekstu na Mowę*.

TEMAT 4

Snake

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- Stosuje zmienne, instrukcje warunkowe, komunikaty i pętle;
- Potrafi wykorzystać klonowanie duszków;
- Stosuje listy do stworzenia tablicy wyników.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafika na wolnej licencji np. pixabay.com.

Przebieg zajęć

Wprowadzenie

Poniższy scenariusz inspirowany jest kultową grą na telefon Nokia 3310 – Snake. Celem tej gry było poruszanie wężem za pomocą klawiszy telefonu (góra, dół, prawo, lewo) i zbieranie pożywienia. Wężem trzeba sterować tak, by nie wpadł na własny ogon, który to wydłuża się po każdym połkniętym kąsku. Wprowadzona w 1997 roku, największą popularność zyskała dopiero w 2000 r., gdy Nokia 3310 była najlepiej sprzedającym się modelem telefonu. Obecnie można znaleźć wersje tej gry w stylu retro na urządzenia z Androidem i Ios.

Wyzwanie 1

Stwórz menu gry. Zaprogramuj klocek *Play*, który będzie rozpoczynał właściwą grę. Dodaj animację przycisku – po najechaniu kursorem myszy klocek powinien się powiększać i zmniejszać. Przygotuj tło gry oraz tytułowego duszka. Głowa i ogon *Węża* mogą być dwoma kostiumami albo osobnymi duszkami. Po uruchomieniu gry i wciśnięciu *Play* głowa węża porusza się po ekranie za pomocą strzałek na klawiaturze.

Wyzwanie 2

Zaprogramuj pojawianie się duszka *Jabłko* w różnych miejscach ekranu. Po zjedzeniu *Jabłka* ogon węża się wydłuża i nalicza się jeden punkt. Wzrost ogona możesz zaprogramować, stosując klonowanie.
Dodatkowym wyzwaniem może być dodanie animacji odliczania do rozpoczęcia gry za pomocą duszka, którego kostiumami są kolejno liczby 3, 2, 1 oraz „start".

Wyzwanie 3

Zaprogramuj sytuację wyjścia *Węża* poza krawędź ekranu oraz wpadnięcia we własny ogon. Dołącz skrypt kończący grę. Możesz dodać dźwięk, zastosować zmianę tła lub dołożyć efekty specjalne np. fajerwerki.

Wyzwanie 4

Zmodyfikuj grę tak, by po skończeniu rozgrywki na ekranie wyświetlał się ranking 5 najlepszych graczy. Każda pozycja powinna składać się z nazwy gracza oraz osiągniętej punktacji. Zadbaj o to, by lista była posortowana malejąco (pierwsze miejsce z najwyższą ilością punktów).

Podsumowanie (wskazówki dla nauczyciela)

Ku inspiracji można pokazać współczesne wersje gry Snake w wydaniu retro.

W sterowaniu duszkiem *Wąż* wykorzystano metodę ustawienia kierunku odpowiedniego do wciśniętej strzałki na klawiaturze, a następnie nieskończonej pętli z instrukcją *przesuń o…*. W tym skrypcie można dodać zabezpieczenie, dzięki któremu *Wąż* nie może pójść w przeciwną stronę, czyli nie ma możliwości obrócić się o 180 stopni (np. idąc w prawą stronę, nie może pójść w lewo, tylko w górę lub w dół).

Wykrywanie ogona można oprogramować za pomocą czujnika *dotyka koloru…* (np. czerwony język lub usta węża).

Ranking punktowy pięciu najlepszych wyników zaprogramowany jest przy użyciu dwóch list: właściwej, zawierającej nazwę gracza i jego wynik, oraz pomocniczej, która odpowiada za posortowanie wyników od największego do najmniejszego. Ten fragment kodu zebrano w nowy blok.

Propozycje modyfikacji dla zaawansowanych uczniów lub gdy dysponujemy większą ilością czasu:

- różne rodzaje jabłek z inną wartością punktową,
- duszki kończące grę,
- szybkość poruszania się węża zależna od długości ogona,
- efekty specjalne na koniec gry np. fajerwerki,
- ilość żyć Węża,
- wersja gry dla dwóch osób.

Notatki do zajęć

TEMAT 5

Planszówka Drabiny i Węże

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- umie poruszać duszkiem po planszy, dostosowując sposób przemieszczania do zdefiniowanych reguł (np. niewychodzenie poza granice planszy, poruszanie się tylko po polach określonego koloru, poruszanie się poziomo/pionowo/zygzakiem/po spirali itp.);
- umie generować wartości losowe z dowolnego zakresu.

JAK PRACUJEMY? (METODY)

- pogadanka,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop.

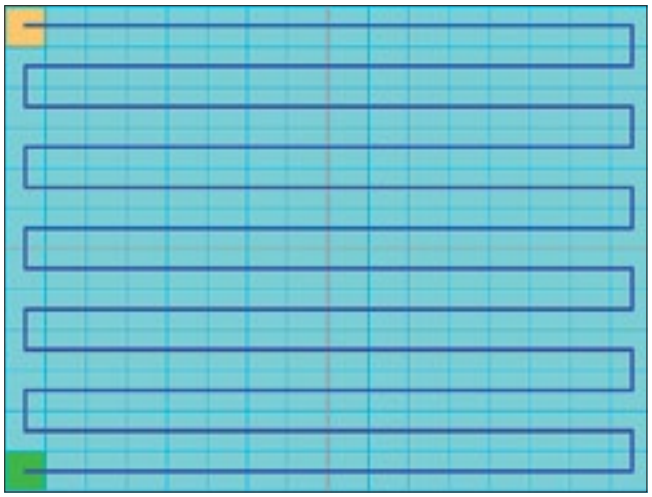
Przebieg zajęć

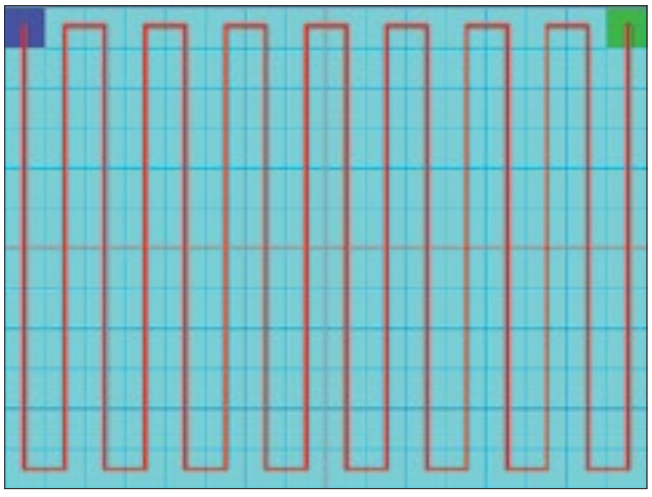
Wprowadzenie

Granie w planszówki jest przyjemnym zajęciem. Obecnie przemysł gier planszowych ma się dobrze – powstaje wiele nowych gier (zarówno rywalizacyjnych, jak i kooperacyjnych) z rozbudowana fabułą, kolorowymi planszami i mnóstwem wymyślnych akcesoriów. Wiele gier znanych od lat doczekało się swoich komputerowych realizacji. Spróbujmy i my stworzyć własną planszówkę na podstawie popularnych gier typu Chińczyk czy *Węże i drabiny*.

Wyzwanie 1

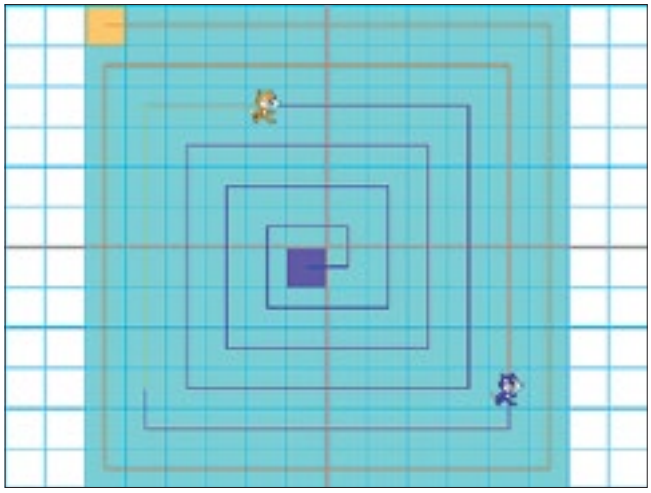
Wybierz planszę do gry (np. kratkowane tło *Xy-grid-30px*). Naucz duszka poruszać się po planszy poziomo/pionowo w taki sposób, by dochodząc do brzegu, zakręcał i szedł dalej, do momentu przejścia całej planszy.





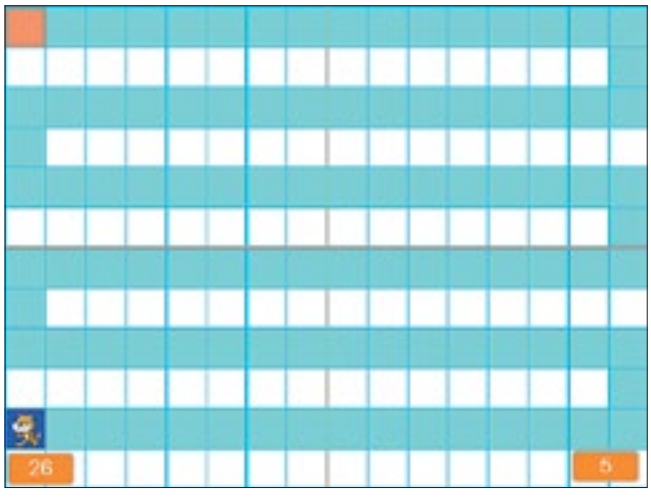
Wyzwanie 2

Wybierz planszę do gry – tak jak poprzednio. Możesz ograniczyć obszar poruszania się duszków do kwadratu (12 na 12 pól). Naucz duszki poruszania się po spirali. Wypróbuj różne położenia początkowe duszków (dowolny róg planszy lub jej środek) oraz różne kierunki (w prawo, w lewo, w górę, w dół).



Wyzwanie 3

Zaprojektuj własną planszę do gry. Ustal reguły poruszania się po planszy (np. przejście od pola startowego do końcowego tylko po polach w wybranym kolorze). Naucz duszka prawidłowo poruszać się po planszy.
Wykorzystaj animację kostki do gry (lub generowanie liczb losowych), aby duszek mógł przemieszczać się o dowolną (zmienną) liczbę pól. Oblicz, w ilu ruchach (po ilu rzutach kostką) duszkowi udało się osiągnąć metę.



Wyzwanie 4

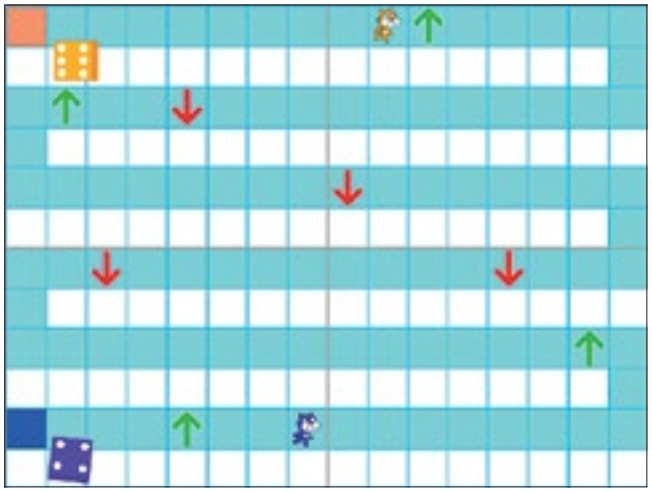
Wykorzystaj planszę oraz skrypty z poprzedniego wyzwania. Zaprojektuj grę, w której dwa duszki poruszają się po planszy. Duszki startują z dwóch różnokolorowych pól znajdujących się na lewym brzegu planszy. Wygrywa ten, kto pierwszy osiągnie metę (pole startowe przeciwnika). Zastosuj animację kostki do gry, by generować liczbę pól, o które przemieszcza się duszek.

Wersja rozszerzona:

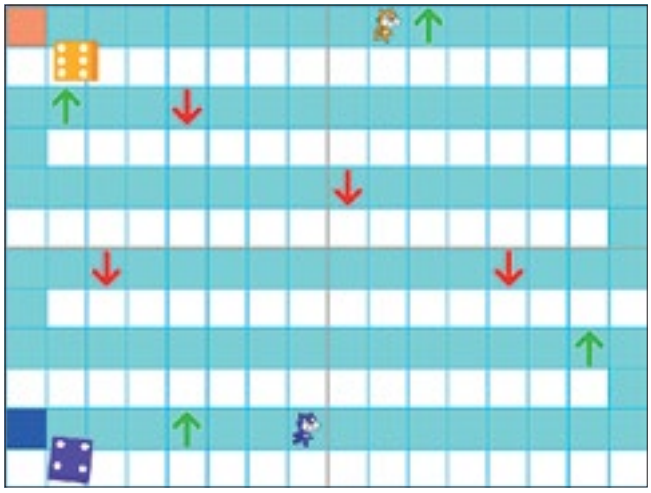
Zaprojektuj grę w Węże i drabiny.

Reguły gry:

- Duszki zaczynają na swoich polach startowych i poruszają się w kierunku pola startowego przeciwnika. Gracze wykonują ruchy naprzemiennie, klikając na swoje kostki, by określić liczbę pól, o które przemieszcza się duszek.



- Po zakończeniu ruchu (przejściu tylu pól, ile wskazywała kostka) duszek zatrzymuje się na wyznaczonym polu.
- Jeżeli duszek zatrzyma się na polu ze strzałką, przenosi się automatycznie o poziom wyżej lub niżej (a w przypadku pierwszej i ostatniej linii – na pole startowe, zgodnie z zasadami przedstawionymi na rysunku).



- Na jednym polu może znajdować się tylko jedna strzałka. Strzałki nie mogą być umieszczone bezpośrednio pod sobą, aby nie powodować zapętlenia się programu.
- Wygrywa gracz, który szybciej dotrze do mety.

Uwagi:

Aby gra była bardziej uniwersalna, strzałki na planszy powinny pojawiać się w losowych pozycjach (z pominięciem zakrętów i pól startowych oraz sytuacji, gdy przeskok w górę lub w dół powodowałby zapętlenie się duszka). Zamiast projektowania statycznych teł lepiej będzie wykorzystać duszki-strzałki, których położenie powinno być losowe, ale zgodne z podstawowymi regułami.

Podsumowanie (wskazówki dla nauczyciela)

Projektowanie planszówki może być dla uczniów i uczennic świetną zabawą. Na początku warto poświęcić trochę czasu na wyćwiczenie prawidłowego przemieszczania się duszków po planszy – w zależności od zastosowanych reguł (wyzwania 1, 2, 3). Młodsi uczniowie mogą stworzyć w ten sposób grę na podobieństwo *Chińczyka*, starsi – wypróbować algorytm szukania drogi w labiryncie. Warto pokazać uczniom, że wykorzystanie zmiennych opisujących rozmiary planszy (lewy/prawy, górny/dolny róg) pozwoli na stworzenie uniwersalnego skryptu dla różnych plansz (np. kwadratów 5x5, 10x10, czy też – mniejszych prostokątów).

Projekt pełnej gry w *Węże i drabiny* można opracować zespołowo, przydzielając różne zadania poszczególnym grupom. Można także zainspirować uczestników i uczestniczki zajęć do tworzenia własnych gier i ustalania bardziej skomplikowanych reguł.

Notatki do zajęć

TEMAT 6

Karciana wojna

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- doskonali stosowanie operacji na listach (dodawanie, wyszukiwanie, usuwanie elementu z listy);
- zna operację dzielenia z resztą;
- umie generować wartości losowe z dowolnego zakresu.

JAK PRACUJEMY? (METODY)

- pogadanka,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- kostiumy kart i przykładowe animacje.

Przebieg zajęć

Wprowadzenie

Gra w karty to rozrywka dla małych i dużych. Na pewno pamiętacie z dzieciństwa grę „w wojnę", którą prowadziliście z dziadkami, rodzicami lub rodzeństwem. Spróbujmy zagrać w karty z komputerem.

Wyzwanie 1

Zapoznaj się z projektami animacji: *link_1* i *link_2*. Stwórz własnego duszka – *TaliaKart* (i zaprojektuj dla niego kostiumy) lub wykorzystaj gotowego duszka z animacji. Napisz skrypt (*intro* do gry), który wyświetli wszystkie karty z talii. Możesz zaprojektować też własną animację.

Wyzwanie 2

Przed grą zawsze tasujemy karty, dlatego napisz skrypt, który pozwoli wyświetlić całą talię kart w losowej kolejności (czyli zasymuluj tasowanie kart).

Wykorzystaj fakt, że kostiumy duszka *TaliaKart* są ponumerowane od 1 do 52. Wygeneruj dwie listy (Gracz_1 i Gracz_2) zawierające wszystkie liczby od 1 do 52 ułożone w losowej kolejności (pamiętaj, że karty nie mogą się powtarzać), następnie wyświetl wszystkie kostiumy duszka *TaliaKart* w kolejności, jaką wskazują te listy (osobno dla Gracza 1 i dla Gracza 2). Zastanów się, jak wyliczyć wartość karty (Walet = 11, Dama = 12, Król = 13, As = 14), znając numer kostiumu.

Wyzwanie 3

Zaprojektuj grę w *Wojnę Karcianą*. Wykorzystaj duszki *Talia_1* i *Talia_2*, aby wyświetlać bieżące karty dla dwóch grających osób. Program powinien informować, który gracz bierze lewę (wygrana Gracza 1 lub Gracza 2) lub (w przypadku równych wartości kart) o wojnie. Wygrana gracza – tak, jak w prawdziwej grze, powoduje, że bierze on wszystkie karty i układa na spodzie talii (na końcu swojej listy). Zastosuj zasadę, że wygrany gracz zbiera najpierw swoje karty, a dopiero potem karty przeciwnika.

Linki zawarte w opisie

link_1 *link_2*

Wyzwanie 4

Zmodyfikuj skrypt rozgrywki w taki sposób, by było możliwe rozstrzygnięcie, kto wygrał wojnę (jedna karta zakryta, nie bierze udziału w rozgrywce, o wygranej lub następnej wojnie decyduje kolejna karta). Gracz, który wygrał wojnę, bierze wszystkie karty (6 w przypadku pojedynczej wojny lub 10 w przypadku wojny podwójnej) i układa je na spodzie talii (na końcu swojej listy), najpierw dodając swoje karty, następnie karty przeciwnika z zachowaniem kolejności, jaka wystąpiła w grze. Napisz program, który pozwoli zakończyć rozgrywkę w jednym z trzech przypadków:

- Jednemu z graczy zabrakło kart (wygrana drugiego gracza);
- Jeden z graczy uzyskał znaczącą przewagę (np. ma o 20 kart więcej niż przeciwnik);
- Gra pozostaje nierozstrzygnięta, ale wykonano już określoną liczbę ruchów (np. 50).

W każdym z tych przypadków program powinien informować o liczbie wykonanych ruchów, rodzaju rozstrzygnięcia (wygrana, przewaga, remis) oraz liczbie kart, które pozostały każdemu z graczy. Można zaprojektować też duszka-przycisk *KoniecGry*, po kliknięciu którego nastąpi natychmiastowe zatrzymanie wszystkich skryptów i wyświetlenie wyników (np. czasu gry, liczby wykonanych ruchów, liczby kart, które pozostały obu graczom, itp.).

Podsumowanie (wskazówki dla nauczyciela)

Pomysł na grę karcianą może zainspirować uczennice i uczniów do tworzenia różnych projektów. Na początku będą to proste animacje (przylot kart, zmniejszanie/powiększanie, obracanie, wirowanie itp.), które dodatkowo uatrakcyjnione odpowiednią muzyką lub efektami dźwiękowymi mogą stanowić wspaniałe intro do gry. Można także zacząć od animacji rozkładania kart (jak w projektach: *link_1* i *link_2*), które będą następnie wykorzystane w pasjansach i innych grach karcianych.

Osobnym problemem jest tasowanie kart, czyli generowanie nieposortowanego ciągu liczb od 1 do n. Dla celów testowych uczniowie i uczennice mogą tworzyć takie ciągi np. w arkuszu kalkulacyjnym, następnie importować dane do list Gracz_1 i Gracz_2. Warto jednak omówić i przetestować (np. pod kątem szybkości działania) różne sposoby generowania nieposortowanego ciągu. Będzie to dobrym ćwiczeniem operacji na listach takich jak: wyszukiwanie, dodawanie na początku/na końcu listy, usuwania duplikatów itp.

Ciekawym rozwiązaniem jest zastosowanie dzielenia z resztą (przez 13), aby przeliczyć numer kostiumu (od 1 do 52) na faktyczną wartość karty. Dla Asa i Króla wynik dzielenia trzeba skorygować (dodać 13), aby wartości tych kart były odpowiednio 14 i 13 (a nie 1 i 0). Wtedy porównywanie „kto wygrał potyczkę" jest w istocie porównaniem 2 liczb, a nie porównywaniem znaków (A, K, D, W).

Rozstrzyganie wygranej w przypadku „wojny" należy dobrze przetestować, aby w sytuacji kilku wojen z rzędu nie zabrakło kart jednemu z graczy (program powinien wtedy zakończyć się natychmiastową wygraną drugiego gracza).

W przypadku grupy zróżnicowanej pod względem wieku lub zaawansowania należy tak rozdzielić zadania (poszczególnym uczniom lub zespołom), by każdy z nich zrobił to, co potrafi najlepiej.

Część uczniów i uczennic może zaprojektować własne karty do gry oraz animacje z muzyką, część prostą rozgrywkę (bez wojny lub z wojną uproszczoną), a troszkę osoby bardziej zaawansowane – rozgrywkę z wojną (mogącą powtarzać się nawet kilkukrotnie, co będzie dobrym przykładem rekurencji). Gotowy projekt powinien uwzględniać różne opcje zakończenia gry (nie tylko taką, że któremuś graczowi zabrakło kart) oraz ciekawą animację końcową dla zwycięzcy.

Warto zwrócić uwagę uczniów na testowanie programu na każdym etapie. Szczególnie ważne są sytuacje graniczne, gdy trzeba rozstrzygnąć, czy następny ruch jest możliwy (brak kart w talii) albo czy nie wystąpił inny warunek końca gry (przekroczony czas, wymagana przewaga jednego z graczy, wymuszenie zakończenia przez użytkownika).

Notatki do zajęć

TEMAT 1

Tajemnicza liczba czy słowo?

LINKI DO PROJEKTU

Wyzwanie 1

Wyzwanie 2

Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- potrafi posługiwać się listami stringów;
- stosuje pętle i warunki;
- tworzy własne bloki;
- stosuje operacje reszty z dzielenia oraz podłogi.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- gra językowa Świńska Łacina.
- strona z grafika na wolnej licencji np. pixabay.com;

Przebieg zajęć

Wprowadzenie

Kody i szyfry istnieją chyba od zawsze... Każdy język jest pewnego rodzaju kodem, którym posługują się obywatele i obywatelki poszczególnych narodów. Zapis matematyczny, działania i liczby to również kodowanie określonych informacji. Kodem posługują się muzycy (nuty), artystki malarki (kolory), tancerze (kroki) oraz oczywiście informatycy (np. język programowania, system binarny itp). Kod czyni informację zrozumiałą dla drugiego człowieka, maszyny czy komputera, szyfr natomiast ma ją utajnić, by nie dostała się w niepowołane ręce.

Poniższe wyzwania wprowadzają w tajniki magicznych liczb i szyfrów oraz wykorzystują zastosowanie list w Scratchu.

Wyzwanie 1

Stwórz program, który będzie scratchową wersją kości „Story Cube" tzn. będzie losował hasła z kilku kategorii (np. postać, miejsce, rzecz), następnie wyświetlał je jednocześnie na ekranie. Przygotuj w notatniku pliki z hasłami – osobny dla każdej kategorii i zaimportuj je do list. Zmodyfikuj skrypt, żeby można było losować hasła dopóty, dopóki użytkownik lub użytkowniczka nie postanowi zakończyć zabawy.

Wyzwanie 2

Napisz program, który zapyta o dzień, miesiąc oraz rok urodzenia, następnie na podstawie tych danych obliczy, jaką liczbą numerologiczną jest użytkownik/użytkowniczka. W kolejnym kroku wyświetli odpowiednią charakterystykę liczby. Dla pełniejszego efektu dodaj duszka, którego kostiumy będą cyframi oraz napisz skrypt wyświetlający kostium przedstawiający obliczoną liczbę numerologiczną.

Wskazówka: Liczba numerologiczna powstaje poprzez sumowanie cyfr daty urodzenia np. data 01.01.2023 roku daje nam wynik 9 (1+1+2+0+2+3=9), a data 31.05.2023 daje wynik 7 (3+1+5+2+0+2+3=16, po czym wykonujemy działanie 1+6=7).

Wyzwanie 3

Napisz program, który pozwala wprowadzić słowo i zakodować je za pomocą szyfru „Świńska Łacina". Szyfr zakłada 2 sytuacje:

- gdy wprowadzone słowo rozpoczyna się samogłoską – wówczas dodajemy na końcu wyrazu „way" np. igła -> igławay;
- gdy wprowadzone słowo rozpoczyna się spółgłoską – wówczas tę spółgłoskę przenosimy na koniec wyrazu i dodajemy „ay" np. piłka -> iłkapay

Wskazówka: Wpisane do zakodowania słowo możesz rozłożyć na litery, które utworzą listę. Pozwoli to na sprawdzenie, czy pierwsza litera jest samogłoską oraz zapamiętanie pierwszej litery w przypadku, gdy słowo rozpoczyna się od spółgłoski.

Efektem końcowym będzie podanie zaszyfrowanego słowa.

Podsumowanie (wskazówki dla nauczyciela)

Wyzwanie pierwsze może być rozgrzewką lub przypomnieniem wybranych elementów pracy z listami w Scratchu. Tematykę tego zadania można dowolnie modyfikować i dopasować w zależności od okoliczności np.:

- życzenia świąteczne/urodzinowe;
- wróżby andrzejkowe/walentynkowe/horoskop;
- zasady bezpieczeństwa w internecie/w trakcie wypoczynku;
- kalambury.

Jeśli nasze uczennice i nasi uczniowie dopiero zaczynają przygodę z listami, należy zachęcić ich do odkrywania różnych sposobów tworzenia list oraz pokazać podstawowe możliwości pracy z nimi.

W zależności od stopnia zaawansowania osób w grupie możemy wybrać, czy realizujemy oba wyzwania – 2 i 3 – czy wybieramy jedno z nich.

W wyzwaniu drugim warto przygotować listę *Przepowiednie*, w której będą zawarte informacje o każdej liczbie numerologicznej i spróbować dostosować czas ich wyświetlania w zależności od długości konkretnego elementu. Można zaproponować, by sumowanie cyfr z daty urodzenia (wyzwanie 2) oraz sprawdzanie, czy wyraz zaczyna się samogłoską (wyzwanie 3) było osobną procedurą, po to by wyrobić w młodych programistach i programistkach nawyk dbania o przejrzystość kodu.

Wyzwanie 3 można rozszerzyć o opcję, w której wprowadzony wyraz zaczyna się od dwóch spółgłosek – wówczas obie spółgłoski przenoszone są na koniec słowa, a dopiero potem dopisana zostaje końcówka „ay".

Notatki do zajęć

TEMAT 2

Bezpieczne Hasła

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- umie korzystać z list – potrafi dodawać elementy do listy oraz przeglądać listę sekwencyjnie;
- potrafi stosować podstawowe operacje na ciągach znaków – wyodrębniać litery, łączyć znaki w słowa, sprawdzać, czy dany znak jest literą bądź cyfrą.

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona do sprawdzanie mocy hasła, np. kaspersky.com.
- strona z grafika na wolnej licencji np. pixabay.com;

Przebieg zajęć

Wprowadzenie

Nie każda informacja może być dostępna dla wszystkich. Chronimy swoją prywatność w sieci, nie udostępniamy niepowołanym osobom ani wrażliwych danych, ani informacji osobistych. Stosujemy hasła dostępu w komputerach, smartfonach i tabletach, aby nikt niepowołany nie przeglądał naszej poczty elektronicznej, nie przejął naszego konta na portalu społecznościowym czy też nie kontrolował naszego komputera. Jesteśmy bezpieczni tylko wówczas, gdy potrafimy zabezpieczyć dostęp do urządzeń i aplikacji tajnymi hasłami, które nawet w razie ataku trudno złamać.

Spróbujmy sprawdzić, czy Scratch może nam pomóc w tworzeniu bezpiecznych haseł.

Wyzwanie 1

Napisz program, który tworzy hasło z kilku podanych słów w ten sposób, że uwzględnia pierwszą literę oraz długość danego słowa.

np. dla ciągu sześciu słów *Kto Się Lubi Ten Się Czubi* hasło brzmi: **K3S3L4T3S3C5**.

Link zawarty w opisie

link_1

Wyzwanie 2

Zastanów się, jakie cechy powinno posiadać bezpieczne hasło. Możesz przetestować moc różnych haseł na stronach zajmujących się badaniem siły hasła (*link_1*).

Napisz program, który pozwoli Ci ustalić reguły bezpieczeństwa (np. długość hasła, minimalną liczbę liter, cyfr i znaków specjalnych), następnie wygeneruje hasło zgodnie z tymi zasadami.

Wyzwanie 3

Wykorzystaj swoje doświadczenia w generowaniu bezpiecznych haseł. Napisz program, który będzie sprawdzał, czy na liście dostępnych haseł wszystkie są poprawne, tzn. utworzono je zgodnie z podanymi regułami.

Listę haseł możesz przygotować w *Notatniku* i zaimportować do programu lub wygenerować losowo za pomocą odpowiedniego skryptu.

Podsumowanie (wskazówki dla nauczyciela)

Zabawy z hasłami są nie tylko pretekstem do porozmawiania z uczennicami i uczniami na temat bezpieczeństwa w sieci, ale mogą też posłużyć jako ciekawe ćwiczenia operacji na znakach i tekstach. W wyzwaniu 1 podano prostą regułę tworzenia hasła. Uczennice i uczniowie mogą także tworzyć własne reguły, które potem ich koledzy i koleżanki będą próbować odgadnąć. Pozwoli to poćwiczyć operacje na ciągach znaków – łączenie znaków, wyszukiwania znaku (wzorca) w tekście, sprawdzanie, czy dany znak jest cyfrą, literą, samogłoską itp., zliczanie znaków określonego rodzaju. Będzie to dobry wstęp do bardziej zaawansowanych programów.

Z uczennicami i uczniami początkującymi możemy wykonać szereg ćwiczeń, które pod pozorem „szyfrowania" nauczą ich swobodnie operować na indeksach elementów i dostrzegać różnicę między wartością elementu (słowo na liście), numerem (indeksem) tego elementu oraz numerem znaku w słowie.

Mogą to być wyzwania w stylu:

- przepisywanie listy słów w odwrotnej kolejności (od końca);
- przeglądanie najpierw elementów o indeksach parzystych, a potem o nieparzystych;
- odwracanie słów od końca do początku (i sprawdzanie czy słowo jest palindromem);
- wybieranie konkretnych liter ze słów (np. ze słowa o indeksie i wybieramy i-tą literę, czyli z pierwszego słowa – pierwszą literę, z drugiego – drugą itd.).

Niestety Scratch nie rozróżnia wielkich i małych liter, nie ma również wbudowanych funkcji takich jak *chr ()* i *ord ()*, co nie pozwala łatwo porównywać ciągów znaków (w Scratchu ciągi *aa*, *aA* oraz *AA* są równe).

Wyzwanie 2 będzie okazją do utrwalenia zasad tworzenia bezpiecznych haseł i stworzenia własnego generatora haseł.

Wyzwanie 3 może być rozbudowaną wersją poprzedniego programu. Z jednej strony użytkownik/użytkowniczka ustala zasady bezpieczeństwa i generuje bezpieczne hasła, z drugiej strony – uczy się testowania programów – czyli jak przygotować dane (losowe hasła), aby sprawdzać wszystkie przypadki szczególne (długość hasła, brak litery/cyfry/znaku specjalnego). Można także zwrócić uczennicom i uczniom uwagę, że przy bardzo długich hasłach (100, a może i 1000 znaków) nie musimy sprawdzać hasła do końca, a jedynie do momentu uzyskania potwierdzenia, że hasło jest poprawne. Pozwala to znacząco skrócić czas wykonywania programu.

TEMAT 3

Szyfry podstawieniowe

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- umie korzystać z list – potrafi dodawać elementy do listy oraz przeglądać listę sekwencyjnie;
- potrafi stosować podstawowe operacje na ciągach znaków – wyodrębniać litery, łączyć znaki w słowa, sprawdzać, czy dany znak jest wybraną literą (np. samogłoską).

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafika na wolnej licencji np. pixabay.com;
- przykłady szyfrów;
- dla chętnych – więcej informacji na temat kryptografii na stronie KhanAcademy.

Przebieg zajęć

Wprowadzenie

Gdy przesyłamy wiadomości otwartym tekstem, jesteśmy narażeni na to, że każdy może ten tekst przeczytać i skorzystać z zawartych w nim informacji. Jeżeli zastosujemy szyfrowanie – informacja będzie nieczytelna dla niepowołanych osób.

Najprostsze szyfry (znane już w starożytności) to szyfry podstawieniowe, w których każdy znak tekstu jawnego zastępowany jest przez inny znak (lub znaki) szyfrogramu. Oczywiście szyfry te są proste do złamania i nie znajdują już zastosowania we współczesnej kryptografii, ale mogą służyć do zabawy i przesyłania koleżankom i kolegom „tajnych wiadomości".

Spróbujmy sprawdzić, czy Scratch może nam pomóc w tworzeniu własnych szyfrów.

Wyzwanie 1

Napisz program, który będzie usuwał z podanych wcześniej słów wszystkie samogłoski.

Słowa można wczytywać pojedynczo, ale lepiej będzie zapisać je na liście wejściowej (DANE), zaś rezultaty działania programu dodawać do listy wyjściowej (WYNIKI). Czy brak samogłosek uniemożliwia nam zrozumienie treści? Co by było, gdyby zamiast usuwać samogłoski – zastąpiono je innymi znakami (np. *)?

Wyzwanie 2

Najprostszym szyfrem jest szyfr podstawieniowy, w którym pewne litery zastępujemy innymi. Np. literę A zastępujemy literą G i na odwrót, literę G zastępujemy literą A. Słowo-klucz powinno zawierać niepowtarzające się pary liter, które zamieniamy między sobą (np. GA–DE–RY–PO–LU–KI, lub MA–LI–NO–WE–BU–TY itp.) Litery, które nie występują w podanym kluczu, przepisujemy bez zmian. Przykład:

Klucz: GA–DE–RY–PO–LU–KI

tekst jawny	M	I	Ł	E	G	O		D	N	I	A
tekst zaszyfrowany	M	K	Ł	D	A	P		E	N	K	G

Napisz program, który zaszyfruje i odszyfruje wiadomość zapisaną szyfrem podstawieniowym:

a. z zastosowaniem klucza GA–DE–RY–PO–LU–KI;
b. z zastosowaniem innego dowolnego klucza.

Wyzwanie 3

Starożytni Izraelici stosowali szyfr podstawieniowy (tzw. Atbasz), w którym zamiana liter odbywała się na zasadzie łączenia w pary liter równoodległych od początku i od końca alfabetu (np. zamiana pierwszej litery z ostatnią, drugiej z przedostatnią itd.). Tabelę kodową dla alfabetu łacińskiego pokazano na rysunku na dole strony.

Można oczywiście zastosować krótszą tabelę:

Pierwsze 13 liter od początku	A	B	C	D	E	F	G	H	I	J	K	L	M
Ostatnie 13 liter od końca	Z	Y	X	W	V	U	T	S	R	Q	P	O	N

Napisz program, który będzie szyfrował wiadomości za pomocą szyfru Atbasz (możesz zamiast alfabetu łacińskiego zastosować alfabet polski).

Tabelę kodową dla alfabetu łacińskiego

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Z	Y	X	V	W	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A

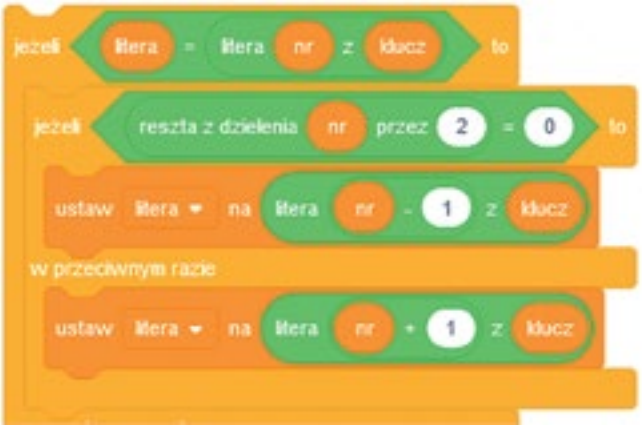
Podsumowanie (wskazówki dla nauczyciela)

Szyfrowanie to ciekawy temat, a równocześnie dobra zabawa. Na początku uczennice i uczniowie mogą ćwiczyć proste operacje na tekstach (np. usuwanie samogłosek lub zastępowanie ich znakami specjalnymi). Początkujące osoby mogą dla szyfrów podstawieniowych wykorzystać proste instrukcje warunkowe, np.:

Warto jednak pokazać, że tak skonstruowany program jest długi (i niezbyt elegancki), a przy tym słabo modyfikowalny.

Wyzwanie 2 pokazuje, że ten sam skrypt może być wykorzystany do różnych szyfrów, niezależnie od wartości i długości klucza.

Operacje na znakach mogą początkującym uczennicom i uczniom sprawić trudność, zwłaszcza, że litery klucza są ustawione parami. Para G—A oznacza zarówno zamianę litery G na A, jak też A na G. Zatem litery z pozycji nieparzystych w kluczu (pierwsza, trzecia, piąta, ...) są zastępowane literami z pozycji „następnej" (druga, czwarta, szósta, ...), zaś litery z pozycji parzystych (2, 4, 6, ...) – literami bezpośrednio je poprzedzającymi (czyli 1, 3, 5, ...).



Aby ułatwić zadanie, można wczytać klucz literka-po-literce do dwóch list

i dokonywać prostej zamiany,

przy czym należy wtedy przeglądać obie listy KLUCZ_1 oraz KLUCZ_2.

Wyzwanie 3 pokazuje, że jeżeli reguły łączenia liter w pary są ściśle określone (pierwsza z ostatnią, druga z przedostatnią itd.), program staje się dużo prostszy (a szyfr nie wymaga podawania klucza).

Notatki do zajęć

TEMAT 4

Szyfr Cezara

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- umie stosować zmienne, instrukcje warunkowe i pętle;
- potrafi stosować podstawowe operacje na ciągach znaków – wyodrębniać litery, łączyć znaki w słowa, sprawdzać, czy dany znak jest wybraną literą;
- umie korzystać z list – potrafi dodawać elementy do listy oraz przeglądać listę sekwencyjnie.

JAK PRACUJEMY? (METODY)

- pogadanka,
- burza mózgów,
- praca indywidualna/ praca grupowa.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- księga szyfrów i szyfr Cezara;
- strona z grafika na wolnej licencji np. pixabay.com;
- alfabet Morse'a (dźwięki)

Przebieg zajęć

Link zawarty w opisie

link_1

Wprowadzenie

Harcerskie szyfry to bardzo często proste szyfry podstawieniowe, w których każdy znak tekstu jawnego zastępowany jest przez inny znak (lub znaki) szyfrogramu. Możemy zastosować nietypową czcionkę z komputera (np. Wingdings), szyfr Czekoladka (link_1) lub spróbować znanego od czasów starożytności szyfru Cezara. Szyfry te są proste do złamania, więc nie znajdują już zastosowania we współczesnej kryptografii, ale świetnie nadają się do zabawy i przesyłania koleżankom i kolegom „tajnych wiadomości".

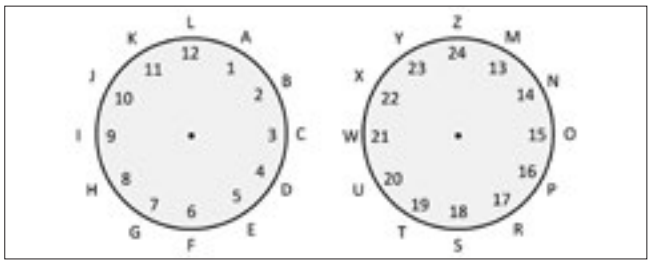
Spróbujmy sprawdzić, jak Scratch może nam w tym pomóc.

Wyzwanie 1

Napisz program, który wykorzysta Szyfr Zegarowy.

Do szyfrowania wiadomości używamy dwóch zegarów. Litery kodujemy zgodnie z rysunkiem – zapisując odpowiadające im godziny, zaś minuty dodając w sposób losowy.

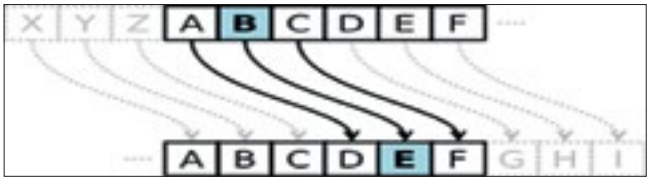
Uwaga: Można zrezygnować z litery X, rzadko używanej w języku polskim, a zamiast niej wprowadzić Ł.



Zaszyfrowaną wiadomość podajemy w postaci listy godzin. KOT to przykładowo:
11:20, **15**:45, **19**:07 (oczywiście minuty nie mają tu żadnego znaczenia, są wprowadzone jedynie, „dla zmyłki").

Wyzwanie 2

Szyfr Cezara jest szyfrem przesuwającym, tzn. każdej literze przyporządkowuje literę oddaloną o określoną liczbę miejsc w alfabecie. Nazwa tego szyfru pochodzi od Juliusza Cezara, który używał go w prywatnej korespondencji do swoich przyjaciół, litery przesuwał o 3. Literę A zastępowano więc literą D, literę B – literą E, zaś litery z końca alfabetu (X, Y, Z) – literami z jego początku (A, B, C).



Napisz program, który dla dowolnego przesunięcia będzie pokazywał sposób przyporządkowania liter. Przykładowo: dla klucza równego 3 koło zewnętrzne pokazuje zaszyfrowaną literę (D), zaś koło wewnętrzne jej odpowiednik (A).

Wyzwanie 3

Napisz program, który pozwoli zaszyfrować i rozszyfrować dowolną wiadomość napisaną szyfrem przesuwającym, czyli tzw. Szyfrem Cezara. Użytkownik powinien mieć wybór długości klucza-przesunięcia oraz możliwość szyfrowania i deszyfrowania wiadomości.

Podsumowanie (wskazówki dla nauczyciela)

Podczas zajęć możemy zachęcać uczennice i uczniów zarówno do wymyślania własnych szyfrów, jak też do wykorzystania znanych algorytmów szyfrujących. Młodsi zapewne zainteresują się alfabetem Morse'a (w wersji dźwiękowej) lub prostymi szyframi (zegarowy, czekoladka). Z bardziej zaawansowanymi grupami możemy oprócz szyfru Cezara omówić także inne znane szyfry.

Podczas omawiania szyfru Cezara warto zwrócić uwagę na wykorzystanie działania *reszta z dzielenia* (operacja modulo), aby litery mogły być zamieniane cyklicznie.

TEMAT 5

System binarny

LINKI DO PROJEKTU

Wyzwanie 1 Wyzwanie 2 Wyzwanie 3 Wyzwanie 4

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- wie, czym jest system binarny i w jakim celu się go stosuje;
- zna algorytm zamiany liczby całkowitej na system binarny i odwrotnie;
- stosuje operacje reszty z dzielenia oraz podłogi;
- tworzy własne bloki, stosuje pętle i operacje na listach.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- pokaz/inscenizacja,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- strona z grafika na wolnej licencji np. pixabay.com;
- opis aktywności offline przy wprowadzaniu systemu binarnego;
- karty do druku.

Przebieg zajęć

Wprowadzenie

W tym scenariuszu będziemy zajmowali się systemami liczbowymi, a dokładniej mówiąc, systemem dziesiątkowym i dwójkowym (binarnym). W dziesiętnym systemie do zapisu liczby używamy cyfr 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Podstawą tego systemu jest 10. Oznacza to, że każdą liczbę możemy zapisać za pomocą kolejnych potęg 10.

np. $185 = 1*10^2+8*10^1+5*10^0$

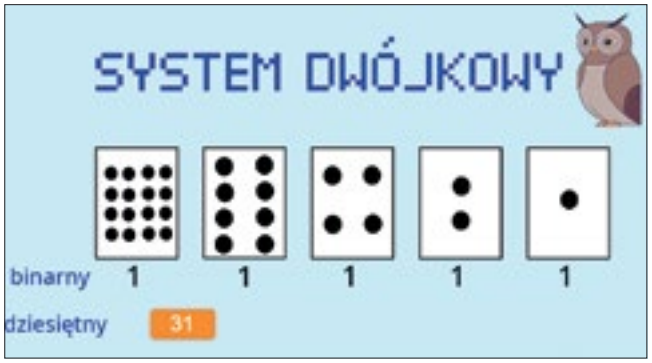
System binarny to sposób zapisu liczby całkowitej oparty o zera i jedynki wskazujące kolejne potęgi liczby 2. System dwójkowy jest używany przez komputery i urządzenia elektroniczne, ponieważ łatwo jest zaprojektować mechanizm, który będzie operował tylko na dwóch cyfrach (zero i jeden). To rozróżnienie dwóch stanów (bitów) oznacza w uproszczeniu „włączony" (1) i „wyłączony" (0) lub „naładowany" (1) i „rozładowany" (0). Poniżej przykład zamiany liczby zapisanej dwójkowo na system dziesiętny.

$11001 = 1*2^4+1*2^3+0*2^2+0*2^1+1*2^0=25$

Więcej informacji można znaleźć na stronie homodigital.pl (*link_1*).

Wyzwanie 1

Napisz program przedstawiający graficznie zamianę liczby z systemu dziesiętnego na binarny. Wykorzystaj karty z kropkami, których liczba na każdej karcie jest równa kolejnej potędze dwójki. Program pozwala klikać w karty i odróżniać dwie sytuacje: karta odkryta – z kropkami (wartość 1) oraz karta zakryta – pusta (wartość 0). Na końcu program powinien wyświetlać liczbę zapisaną w systemie dziesiętnym oraz dwójkowym.



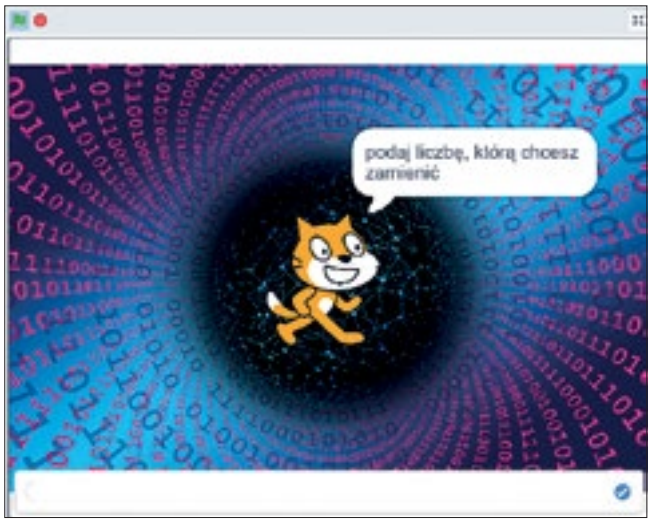
Wyzwanie 2

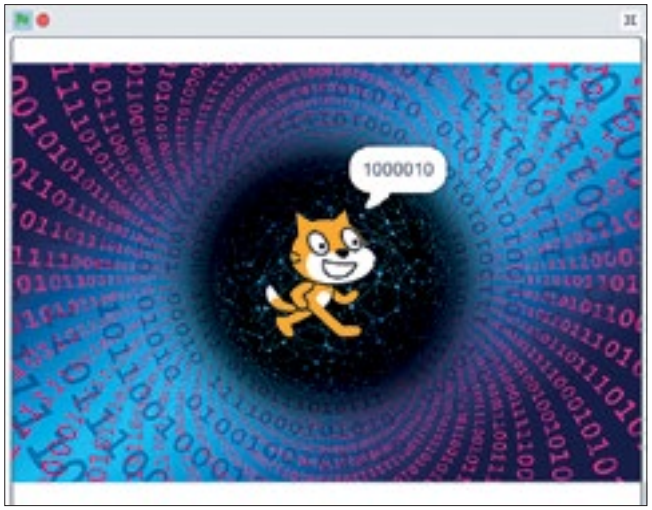
Stwórz program, który po kliknięciu będzie zapalał lub gasił 8 ułożonych w szeregu żarówek. Każda kolejna żarówka odpowiada kolejnej potędze dwójki (począwszy od wykładnika 0). W zależności od stanu żarówki powinna być wyświetlona informacja o wartości 1, gdy żarówka jest zapalona, i wartości 0, gdy żarówka jest wyłączona. W efekcie na ekranie powinien pojawić się ciąg zer i jedynek.

Następnie zmodyfikuj program tak, by użytkownik mógł policzyć, jaka liczba dziesiętna kryje się za utworzonym kodem binarnym. Program powinien sprawdzić, czy wpisana odpowiedź jest prawidłowa i wyświetlić odpowiedni komunikat. W przypadku niewłaściwej odpowiedzi powinien pozwolić na udzielenie innej odpowiedzi.

Wyzwanie 3

Napisz program zamieniający liczbę dziesiętną (wpisaną przez użytkownika/użytkowniczkę) na system binarny. Wynik powinien zostać wypisany na ekranie.





Wyzwanie 4

Stwórz kalkulator umożliwiający zamianę liczby dwójkowej na dziesiętną i odwrotnie. W tym celu zmodyfikuj program z Wyzwania 3, dodając konwersję liczby binarnej na dziesiętną. Zaprogramuj dwa przyciski pozwalające wybrać sposób konwersji liczby. Zadbaj o przejrzystość kodu i stwórz bloki odpowiadające za przekształcenie liczb w każdą stronę.

Linki zawarte w opisie

link_1 *link_2* *link_3*

Podsumowanie (wskazówki dla nauczyciela)

Z uczniami i uczennicami rozpoczynającymi naukę systemu dwójkowego warto przeprowadzić aktywność offline opisaną na stronie csunplugged.org (*link_2*, *link_3*). Pozwoli to nawet młodszym dzieciom, które nie znają pojęcia potęgi, zrozumieć zasadę zapisu liczby dziesiętnej w postaci zer i jedynek.

Wyzwania 1 i 2 pokazują obrazowo, na czym polega konwersja liczby na system binarny. Nauczyciel/nauczycielka może wybrać, którą wersję graficzną chce z uczniami i uczennicami zrealizować.

Wyzwanie 3 jest implementacją algorytmu zamiany liczby zapisanej w systemie dziesiętnym na liczbę binarną. W dołączonym skrypcie wykorzystano listy, ale można stworzyć skrypt bez ich użycia.

Wyzwanie 4 jest kontynuacją Wyzwania 3 poszerzoną o konwersję w drugą stronę, czyli zamianę liczby binarnej na liczbę w systemie dziesiętnym.

Powstały kalkulator zamieniający liczby binarne na dziesiętne i na odwrót można rozszerzyć o inne systemy liczbowe (ósemkowy, szesnastkowy) oraz dodać animacje, dźwięki, zamianę tekstu na mowę itp.

Notatki do zajęć

TEMAT 6

Czy jestem palindromem?

LINKI DO PROJEKTU

Wyzwanie 1

Wyzwanie 2

Wyzwanie 3

DO CZEGO DĄŻYMY? (CELE)

Po zajęciach uczeń/uczennica:

- tworzy własne bloki, stosuje pętle i operacje na listach;
- wyodrębnia litery wprowadzonej frazy oraz łączy je w słowa;
- wykorzystuje rozszerzenie Zamiana tekstu na mowę;
- optymalizuje swój program.

JAK PRACUJEMY? (METODY)

- praca indywidualna,
- burza mózgów.

JAK PRZYGOTOWAĆ SIĘ DO ZAJĘĆ? (POTRZEBNE MATERIAŁY, ZASOBY, ADRESY)

- strona scratch.mit.edu lub Scratch3.0 w wersji desktop;
- artykuł o wyjątkowych datach – palindromach.
- strona z grafika na wolnej licencji np. pixabay.com;

Przebieg zajęć

Wprowadzenie

Szyfrowanie służy do zachowania poufności danych. Najprościej rzecz ujmując, informacje są zniekształcane tak, żeby niepowołane osoby nie mogły ich odczytać. System szyfrowania często składa się z dwóch programów komputerowych: pierwszy służy do zaszyfrowania danych (nazywanych tekstem jawnym), przekształcając je w nieczytelny i niezrozumiały komunikat (szyfrogram), a drugi program służy do odszyfrowania szyfrogramu, zmieniając go z powrotem w tekst jawny. Poniższe wyzwania przybliżają nam szyfr odwrócony. Najpierw zajmiemy się „odwróconym" czytaniem, czyli czytaniem wyrazów od końca np. mamo –> omam. Następnie poznamy palindromy, czyli wyrażenia, które czytane zarówno od lewej, jak i od prawej brzmią tak samo.

Wyzwanie 1

Stwórz projekt odczytujący wspak wpisane słowa lub frazy. Wykorzystaj rozszerzenie *Zamiana tekstu na mowę*.

Wyzwanie 2

Napisz program sprawdzający, czy wprowadzona przez użytkownika/użytkowniczkę liczba, słowo lub zdanie jest palindromem. Zaprogramuj możliwość wielokrotnego wpisywania liczb, słów i wyrażeń.

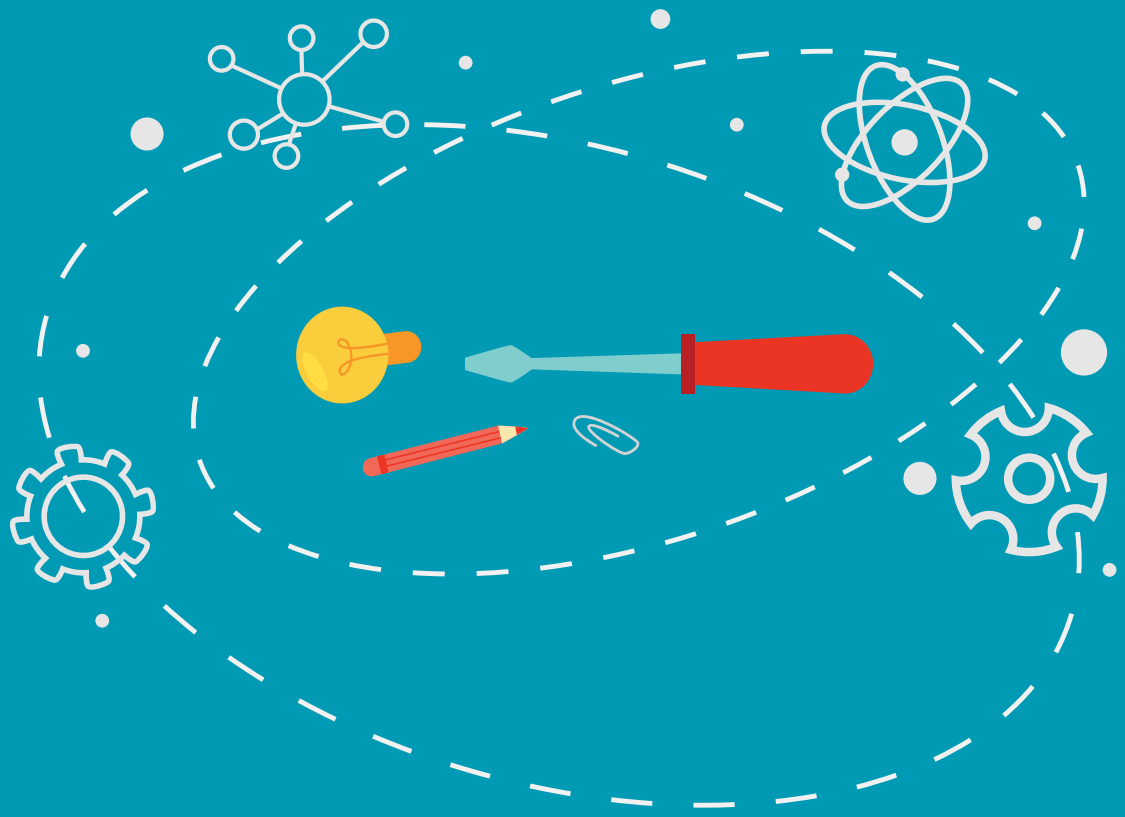
Wyzwanie 3

Napisz program, który wczytuje listę słów/zdań i dla każdego z nich określa, czy był to palindrom. Jeżeli zdanie składa się z kilku słów, to spacje między nimi zignoruj.

Podsumowanie (wskazówki dla nauczyciela)

Wyzwanie 1 można potraktować jako rozgrzewkę wykorzystującą rozszerzenie Scratcha *zamiana tekstu na mowę*. Każda litera wpisanego słowa trafia w odwróconej kolejności na listę „wspak". Pomysł ten wykorzystano w realizacji kolejnego zadania. Zaproponowane rozwiązanie wyzwania drugiego jest mało optymalne. Algorytm polega na rozbiciu na litery wpisanej przez użytkownika/użytkowniczkę frazy i dodanie ich do listy *palindrom*, przy czym pierwsza litera frazy zostaje umieszczona na ostatniej pozycji listy, druga litera na przedostatniej itd. Dalej następuje porównanie frazy z listą i wyprowadzenie odpowiedzi. Taka koncepcja jest dobra dla pojedynczych (krótkich) słów, ale jeśli słowa są długie (i na dodatek jest ich dużo – jak w wyzwaniu 3), to należy zoptymalizować program i porównywać ze sobą pary liter (pierwszą z ostatnią, drugą z przedostatnią itd.), a w przypadku pierwszej niezgodności kończyć dalsze porównywanie. W celu szukania całych zdań, które są palindromami (np. Kobyła ma mały bok), wprowadzono funkcję, która usuwa spacje.

Dodatkowym wyzwaniem może być napisanie programu, który sprawdza, czy data zapisana w formacie DD-MM-RRRR jest palindromem np. 22.02.2022. Takich dat w bieżącym stuleciu jest zaledwie 38, ale jeśli rozszerzymy rozważania na inne formaty zapisu dat (D-M-RR, M-D-RR, M-DD-RRR, M-DD-RR), znajdziemy znacznie więcej przykładów.



Centrum
Mistrzostwa
Informatycznego

www.cmi.edu.pl