Progr@muj w zespole

Materiały dla nauczycieli

Materiały dla nauczycieli – spis treści

Scenariusz zajęć

Scenariusz zajęć

Rozwijamy umiejętność współpracy w zespole w trakcie zadania programistycznego. Obszar tematyczny: informatyka

Etap edukacyjny

Zajęcia zaplanowano dla uczniów klas II – III liceów ogólnokształcących, uczęszczających do klas z rozszerzonym programem z zakresu informatyki, a także dla uczniów klas II – IV techników, uczących się w zawodzie technik – informatyk lub technik programista. Nie ma również przeciwwskazań, abyście zastosowali scenariusz w odniesieniu do uczniów klas I szkół średnich pod warunkiem, że będą oni posiadali niezbędną wiedzę i umiejętności z zakresu programowania, minimalnie:

- podstawowa wiedza o konstrukcjach programistycznych
- znajomość jakiegokolwiek języka programowania tekstowego (C; Pascal; LOGO; BASIC; JavaScript; itp.) lub znajomość Python na poziomie podstawowym

Dodatkowo przydatna będzie znajomość języka angielskiego na poziomie średnim (konieczność czytania dokumentacji technicznej w języku angielskim, choć można korzystać z serwisów tłumaczących).

Informacje organizacyjne

Grupa uczestników (klasa) podzielona zostaje na grupy 4 – osobowe (ostatnia grupa między 2 a 5 os.) na podstawie przeprowadzonego na wcześniejszych zajęciach (lekcja "0") prostego testu predyspozycji i ról w zespole.

Poziom zaawansowania uczniów w zakresie znajomości języka Python: podstawowy. Zakładamy w naszych samouczkach podstawowy poziom znajomości zagadnień związanych z programowaniem w ww. języku, ale młodzież może do niego dojść samodzielnie, bazując na ogólnodostępnych kursach, np.:

- Kurs: https://it-szkola.edu.pl/kkurs,kurs,216 lub
- Kurs: https://it-szkola.edu.pl/kkurs,kurs,217 lub
- https://python.szkola.pl/szkolenia-python-edu/python101x-edu/

Treści nauczania

Tematyka i program zajęć odwołują się do podstawy programowej z rozszerzonego zakresu informatyki:

a) Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi (cel kształcenia nr II);

b) Rozwijanie kompetencji społecznych, takich jak: komunikacja i współpraca w zespole, w tym w środowiskach wirtualnych, udział w projektach zespołowych oraz zarządzanie projektami (cel kształcenia nr IV).

Cele edukacyjne dla uczniów (efekty uczenia się)

Dzięki uczestnictwu w zajęciach w zakresie kompetencji programistycznych uczeń:

Wiedza

a. zna podstawowe zasady efektywnej komunikacji interpersonalnej;

b. wie jakie czynniki determinują sukces w pracy zespołu;

c. umie wykorzystać ogólnodostępne API;

d. wie jak posługiwać się środowiskiem programistycznym PyCharm;

e. potrafi odróżnić różne elementy języka programowania.

Umiejętności

a) potrafi sformułować klarowny komunikat i sprawdzić czy został on właściwie zrozumiany;

b) radzi sobie z sytuacją konfliktową w zespole;

c) potrafi korzystać z serwisu GitHub do współpracy z innymi programista-

mi;

d) radzi sobie z systemem operacyjnym Linux;

e) realizuje obiektowe paradygmaty programowania.

Postawy / kompetencje społeczne

a) zachowuje się odpowiedzialnie w stosunku do pozostałych członków zespołu i wywiązuje się z własnych zobowiązań, mając na celu osiągniecie wspólnego celu zespołu;

b) reaguje ciekawością na odmienne opinie i poglądy, rozwiązując konflikty w trybie porozumienia;

c) traktuje inne osoby z szacunkiem dla ich odmienności;

d) komunikuje się w sposób klarowny, wykorzystując narzędzia efektywnej komunikacji interpersonalnej;

e) zyskuje świadomość swoich własnych zasobów: umiejętności, talentów i mocnych stron.

Czas trwania (liczba godzin): 7 h lekcyjnych (zajęcia szkolne) + 5 h zegarowych (praca po zajęciach)

- 1 x 45 zajęcia wprowadzające, wyjaśnienie zasad, jakie będą obowiązywać w trakcie zajęć oraz ról pełnionych przez nauczyciela i uczniów, zawarcie kontraktu z uczniami, pre-test kompetencji współpracy i komunikacji w zespole oraz test predyspozycji i ról w zespole.
- 5 x 45 minut zajęć stricte programistycznych
- 5 x 60 minut pracy samodzielnej uczniów (w ramach stworzonego zespołu - tu ilość czasu zależy od predyspozycji członków zespołu)
- 1 x 45 minut spotkania podsumowującego, końcowy test kompetencji współpracy i komunikacji w zespole

Metody i formy pracy

- metoda podawcza (wprowadzenie + filmy wspomagające)
- metoda projektu (praca w zespołach w czasie lekcji i pozalekcyjna)

- lekcja odwrócona (znaczna część materiału będzie przyswajana w domu, zaś praca nad projektem zespołowym w dużej mierze ma odbywać się podczas zajęć szkolnych)

Środki dydaktyczne

filmiki (pigułki wiedzy)

- krótki przewodnik po zajęciach dla uczniów
- przykładowe kody źródłowe na platformie GitHub
- instrukcje dot. poszczególnych lekcji wspomagające nauczycieli
- przewodnik po kompetencjach współpracy i komunikacji w zespole dla nauczycieli
- testy wspomagające realizację zajęć (p. przebieg zajęć).

Wymagania techniczne

Do prowadzenia zajęć niezbędne są:

- *komputery z procesorem minimum Core 3, 4 GB RAM dla uczestnika zajęć; **nie można** przeprowadzić zajęć z wykorzystaniem tabletów/smartfonów*
- *system operacyjny: Linux lub MacOS lub MS – Windows*
- *zainstalowane oprogramowanie PyCharm Community oraz Python w wersji min. 3.8, LibreOffice, GIMP, Bluefish (wszystkie to Open Source dostępne na wszystkie systemy operacyjne)*
- *połączenie z siecią Internet łączem min. 100 MBit (w praktyce – dowolnym połączeniem zapewniającym płynne odtwarzanie wideo dla celów pokazu)*
- *projektor/monitor oraz nagłośnienie niezbędne do odtwarzania wybranych odcinków wideotutorialu przez nauczyciela w trakcie zajęć*

W celu unifikacji i ułatwienia pracy przygotowaliśmy kompletny system operacyjny Linux Ubuntu 20.04 LTS w wersji maszyny wirtualnej OVA dla środowiska Virtualbox. Więcej informacji znajduje się w instrukcjach dla nauczycieli.

Nasz projekt oparliśmy o programowanie w języku Python 3.8 z wykorzystaniem dodatkowych bibliotek (PySimpleGUI, requests), które są ogólnodostępne. Stworzona aplikacja będzie możliwa do uruchomienia tylko na komputerze z zainstalowanym minimum Python 3.6 . Dodatkowo wykorzystujemy też inne aplikacje, wszystkie na otwartych licencjach:

- edytor tekstów Libre Office Writer (https://libreoffice.org)
- edytor grafiki GIMP (https://gimp.org)
- edytor HTML Bluefish (https://bluefish.openoffice.nl)

Przebieg zajęć

Zaproponowany przez nas podział materiału można rozpisać na następujące jednostki dydaktyczne:

0. Lekcja w szkole - wprowadzenie, zasady pracy, testy wstępne,

1. Lekcja w szkole - ustawienia wstępne środowiska

* Praca samodzielna - maszyna wirtualna i podstawy aplikacji

2. Lekcja w szkole - podstawowe typy danych i konstrukcje programistyczne w Python

* Praca samodzielna - obsługa głównych elementów biblioteki PySimpleGUI

3. Lekcja w szkole - requests i API - słowniki i JSON

* Praca samodzielna - samodzielne testy dostępu do API

4. Lekcja w szkole - różne interfejsy aplikacji, `Commit/Push` do repozytorium

* Praca samodzielna - definiowanie funkcji w Python i dalsze przygotowywanie dokumentacji

5. Lekcja w szkole - praca z kluczami i wartościami słowników

* Praca samodzielna - końcowe tworzenie dokumentacji

6. Lekcja w szkole - podsumowanie projektów, wybór najlepszego projektu, post-testy itp.

Lekcja nr 0 (45 min.)

Lekcja ma na celu wprowadzenie uczniów w kontekst zajęć, ich cele, planowany przebieg, a także przekazanie źródeł, na podstawie których uczniowie powinni przygotować się do zajęć (przewodnik dla uczniów, materiały wideo), przeprowadzenie dwóch testów: prostego testu predyspozycji i ról w zespole służącego przygotowaniu podziału na zespoły (5 minut) oraz testu umiejętności współpracy i komunikacji w zespole, który stanowił będzie punkt odniesienia dla oceny nabycia kompetencji społecznych w wyniku przeprowadzonych zajęć (10-15 minut). Nauczyciel i uczniowie ustalają zasady pracy podczas zajęć, tzw. kontrakt (więcej informacji dot. zasad pracy, podziału, kontraktu w *Przewodniku dla nauczycieli*). Nauczyciel może również zaproponować, że zwycięskie/wszystkie zespoły, które ukończyły i oddały działające aplikacje wraz pozostałymi materiałami (strona internetowa etc.) otrzymają oceny celujące. Zawsze proponujemy tylko nagrodę, aby motywować uczniów do podjęcia wyzwania.

Pomiędzy lekcją 0 i lekcją 1 nauczyciel przeprowadza podział na zespoły na podstawie przeprowadzonego wcześniej testu predyspozycji, swojej znajomości poziomu zaawansowania uczniów oraz wg parytetu płci. Informacja o podziale na zespoły jest przekazywana uczniom co najmniej na 1 dzień przed rozpoczęciem właściwych zajęć. Następnie uczniowie kontaktują się ze sobą i wybierają temat pracy swojego zespołu.

Zespoły pracują nad wybranym przez siebie zadaniem samodzielnie. Członkowie zespołu dzielą pracę między siebie w taki sposób, że każdy z nich ma do wykonania swoje zadania i jest za nie odpowiedzialny. Podczas pracy zespołowej nauczyciel jest jedynie wsparciem i, w razie potrzeby, mediatorem.

Warto zaznaczyć, że celem projektu jest wspólna praca nad projektem, poznanie języka programowania, narzędzi edycyjnych; filmy podzielone są na te do wyświetlania w szkole (gdzie ważne jest wsparcie nauczyciela) i do obejrzenia samodzielnie (aby wyrobić umiejętności samodzielnej nauki).

Lekcja nr 1 (45 min.)

Rozpoczyna się od ankiety wśród uczniów - jakie systemy operacyjne znają, czy wiedzą, na które z nich dostępny jest Python, Libre Office, GIMP, Geany, czy znają różne licencje, w tym OpenSource. Przede wszystkim należy skupić się na instalacji odpowiednich aplikacji, dla ujednolicenia interfejsu proponujemy skorzystać z rozwiązania OVA dla VirtualBox. Jeśli nie korzystamy z OVA to w systemie Windows pobieramy instalatory ze stron internetowych:

- https://www.python.org/ftp/python/3.8.10/python-3.8.10-amd64.exe
- https://www.libreoffice.org/download/download/
- https://www.gimp.org/downloads/
- https://geany.org/download/releases

Po instalacji aplikacji uczniowie tworzą pierwszy projekt w środowisku PyCharm i w nim wykonują pierwszy minimalny program - okno aplikacji z komunikatem, np. "Hej - witamy w zespole!". W ten sposób poznają środowisko pracy i od razu widzą efekt swojej pracy. Przykładowy kod źródłowy znajduje się w repozytorium GitHub (https://github.com/programujemy-python/programuj-w-zespole-test) w katalogu spotkanie_1/00_start

Następnie tworzą swoje konta w serwisie GitHub i poznają różnego rodzaju licencje na materiały i oprogramowanie, zapoznają się też z podstawami pracy w edytorach wykorzystywanych w naszym projekcie.

W trakcie pracy samodzielnej uczniowie zapoznają się z filmami omawiającymi importowanie maszyny OVA i podstawy tworzenia dokumentacji, grafiki, stron internetowych oraz ustalają pomiędzy sobą szczegóły wykonania projektu (np. nazwa aplikacji, czy rodzaj informacji, z których chcą korzystać).

Lekcja nr 2 (45 min.)

Poświęcona jest omówieniu tych elementów i konstrukcji programistycznych, które będą niezbędne w dalszych etapach prac, a więc:

- podstawowych typów danych w Python i konwencji F-String;
- zaawansowanych typach danych: listach, słownikach;
- sposobach importowania modułów zewnętrznych;
- pętlach iteracyjnych;
- instrukcjach warunkowych.

Dodatkowo omawiamy:

- listy w edytorze tekstów
- warstwy w edytorze graficznym
- kontenery w edytorze HTML

W trakcie pracy samodzielnej uczniowie zapoznają się z filmami dotyczącymi zasad działania biblioteki PySimpleGUI oraz kolejnymi elementami dokumentów tekstowych, grafik czy stron WWW.:

- tworzenie layoutu aplikacji;
- sterowanie zdarzeniami;
- wyświetlaniem informacji;
- dodawaniem elementów sterujących (przycisków);
- dodawaniem obrazków;
- sposobami wprowadzania danych.

Na tym etapie mogą próbować tworzyć szkielet aplikacji, czyli tworzyć okna z elementami najważniejszymi z punktu widzenia funkcjonalności. Osoby odpowiedzialne za dokumentację, grafikę i HTML mogą zacząć

tworzenie odpowiednich elementów, jak dokumentacja, strona WWW reklamowa, grafiki do projektu.

Lekcja nr 3 (45 min.)

Poświęcona jest na analizowanie systemu API, z którego zespoły wybierają elementy do swoich projektów. Uczniowie wykorzystują Python Console w PyCharm, sprawdzają dokumentację różnych przykładowych API. Uczą się wykorzystywać moduł requests do komunikacji z API. Dzięki temu otrzymują odpowiedzi w formacie JSON, które analizują wykorzystując pętle iteracyjne "for". Tworzą pliki PDF z dokumentacją czy PNG z własnymi grafikami. W tej lekcji niezbędne jest działające łącze internetowe.

W trakcie pracy samodzielnej uczniowie testują samodzielnie różne API, aby zapoznać się z rodzajami zwracanych informacji, testują błędy, jakie mogą wystąpić podczas używania API. Tworzą własne repozytorium w serwisie GitHub. Od kolejnej lekcji mogą swoją pracę wysyłać do serwisu GitHub.

Lekcja nr 4 (45 min.)

W trakcie tego spotkania poznajemy sposoby definiowania funkcji i przestrzenie w Python (ang. `namespace`); zapoznamy się ze skryptem odczytującym i prezentującym dane z przykładowego API. W drugiej części pracujemy ze zdalnym repozytorium GitHub.

- definiowanie funkcji
- koncepcja przestrzeni nazw w Python
- praca ze zdalnym repozytorium GitHub

W trakcie pracy samodzielnej uczniowie sprawdzają dane odczytywane z API za pomocą skryptu wysyłają też projekt lokalnie zapisany do repozytorium GitHub.

Lekcja nr 5 (45 min.)

Przeznaczona jest na utrwalenie wiedzy o słownikach, modyfikacji ich elementów. Możemy też wykorzystać ten czas dla uczniów na ostatnie zapytania. W trakcie pracy samodzielnej uczniowie kończą tworzenie aplikacji, dokumentacji i strony internetowej Tu nie przewidujemy już żadnych filmów. Dla zespołów najsłabszych przewidzieliśmy przykładowy kod aplikacji, przykładowy dokument tekstowy dokumentacji i stronę internetową opisującą

aplikację.

Lekcja nr 6 (45 min.)

Przeznaczona jest na podsumowanie projektów; zespoły prezentują wyniki swojej pracy, mogą też wzajemnie komentować swoje aplikacje. Podczas tej lekcji przeprowadzamy też ponownie test umiejętności współpracy i komunikacji w zespole. Przy komentowaniu i sprawdzaniu prac możemy posłużyć się formatterem kodu Black (https://black.readthedocs.io/en/stable/) i sprawdzić, ile zmian wykona.

Sposób oceny osiągnięcia celów

- nauczyciel na zakończenie ocenia, czy dany zespół uczniowski zrealizował swój projekt, tj. osiągnął zamierzony cel, przygotowana prosta aplikacja działa poprawnie (zgodnie z intencją zespołu), a kod jest poprawny i dobrej jakości
- druga składowa oceny dotyczy poprawy umiejętności współpracy i komunikacji w zespole w stosunku do wartości bazowej (pre-test) - weryfikacja następuje z wykorzystaniem narzędzia udostępnionego nauczycielowi w ramach naszego modelu (test umiejętności współpracy i komunikacji w zespole) - wynik testu poprawiony o min. 30% w stosunku do wartości bazowej dzięki udziałowi w zajęciach uznaje się za w pełni satysfakcjonujący.

Oceny stopnia osiągnięcia celów nie należy mylić z ocenami szkolnymi - przed lekcją 0 nauczyciel powinien zdecydować, czy za wykonane zadania będą przyznawane oceny (rekomendujemy motywowanie przez nagrodę, np. ocenę celującą za ukończenie projektu, tj. osiągnięcie opisanego powyżej celu zajęć w dwóch wymiarach).

Materiały pomocnicze:

1) Materiały wspierające dla uczniów i nauczycieli z zakresu programowania:

Kurs: https://it-szkola.edu.pl/kkurs,kurs,216 wraz z kodami w serwisie GitHub: https://github.com/klubmlodegoprogramisty/python/tree/main/poziom_podstawowy

- Kurs: https://it-szkola.edu.pl/kkurs,kurs,217 wraz z kodami w serwisie GitHub: https://github.com/klubmlodegoprogramisty/python/tree/main/poziom_sredniozaawansowany
- Kurs: https://python.szkola.pl/szkolenia-python-edu/python101x-edu/
- Dokument: https://kodujwpythonie.pl/Dodatek_A_GIT_A4_.pdf

Książki:

- A. Jurkiewicz, Python 3. Projekty dla początkujących i pasjonatów. Wyd. Helion, Warszawa 2022,
- Konrad Jagaciak, PYTHON, kurs programowania na prostych przykładach, ringier Axel Springer, Warszawa 2019,
- Sean McManus, Misja Python, Utwórz swoją kosmiczną grę! PWN, Warszawa 2019,
- Marek Luliński & Gniewomir Sarbicki, Python, C++, JavaScript. Zadania z programowania.Wyd.Helion, Warszawa 2018,
- Michał Wiszniewski, Python na start! Programowanie dla nastolatków. Wyd. Helion, Warszawa 2017,
- A. Jurkiewicz, K. Zadrzyński, K. Wasilkowska, M. Trojanowicz, Koduj w Pythonie. Tworzymy grę przygodową. Wyd. Fundacja Rozwoju Edukacji Cyfrowej, Gdańsk 2020.

2) Materiały wspierające dla uczniów i nauczycieli w zakresie komunikacji i współpracy w zespole:

- J. Kołodziejczyk, K. Salamon-Bobińska, N. Karaszewski, S. Bobula, Nauczanie kooperatywne w: Edukacja jako odpowiedź. Praca zbiorowa pod red G. Mazurkiewicz, Warszawa 2014;
- M. Spławska- Murmyło, A. Wawryszuk, Współpraca zespołowa z wykorzystaniem TIK; Materiały ORE:http:bc.ore.edu.pl. pobór 5/1/2022. Warszawa 2017;

Książki:

- M. Adams, Myślenie pytaniami. Wyd. Studio EMKA, Warszawa 2020,

- S. Stahl, Jak się dogadywać mimo różnic. Instrukcja obsługi typów osobowości. Wyd. OTWARTE, Kraków 2021,
- R. Gut, A. Gut, Pokolenie Y. Zarządzanie sobą na Zielonej Ścieżce. Wyd. Instytut Flashpoint, Wrocław 2016,
- J. Lamri, Kompetencje XXI wieku. Wyd. Wolters Kluwer, Warszawa 2021,
- T. Kozłowski, Zanim będzie za późno. O potrzebie rozwoju kompetencji społecznych w edukacji. Wys. Wolters Kluwer, Warszawa 2020,
- James M. Heidema. Carol A. MCKenzie, Budowanie zespołu z pasją. Dom Wydawniczy Rebis, Poznań 2006,
- M. Płócińska, H. Rylke, Czas współpracy i czas zmian. Wyd. WSiP, Warszawa 2002,
- E. Góralczyk, Umowa z klasą. Wyd. Fraszka Edukacyjna, Warszawa 2015

Autorzy scenariusza:

1) Adam Jurkiewicz
2) Rafał Kamiński
3) Konrad Kosieradzki
4) Elżbieta Piotrowska-Gromniak

Instrukcja dla nauczycieli

Instrukcja lekcji dla nauczycieli.

Temat: Lekcja 0 – wprowadzenie w projekt.

Autorzy: Konrad Kosieradzki, Adam Jurkiewicz, Rafał Kamiński

Opis ogólny:

W trakcie tego spotkania uczniowie otrzymają wstępne informacje nt. celu głównego zajęć, ich zakresu tematycznego, powiązania z podstawą programową, dowiedzą się, jakie materiały są do ich dyspozycji, w jaki sposób ma wyglądać ich praca na lekcjach i poza nimi, a także wykonają testy (wypełnią kwestionariusze), które umożliwią dokonanie podziału na zespoły opracowujące projekty programistyczne oraz pozwolą zmierzyć wejściowy poziom kompetencji komunikacji i współpracy w zespole poszczególnych uczniów.

Podstawowe treści omawiane w materiale:

Nie dotyczy - lekcja ma charakter wprowadzająco-organizacyjny.

Przebieg lekcji

Czynności przygotowawcze przed lekcją:

1. Zapoznaj się ze Scenariuszem zajęć, niniejszymi instrukcjami lekcji oraz Przewodnikiem dla nauczycieli.

2. Zapoznaj się z zawartością filmów przygotowanych dla uczniów, dotyczących programowania w języku Python, pozostałych umiejętności okołoprogramistycznych (HTML, grafika, dokumentacja), a także komunikacji i współpracy w zespole. Pozwoli Ci to lepiej zaplanować przebieg zajęć, ew. uzupełnić wiedzę i wypełnić rolę tutora, wspierającego uczniów w realizacji projektów zespołowych.

3. Przygotuj *Test umiejętności współpracy i komunikacji w zespole* oraz *Test* predyspozycji i ról - szablony możesz pobrać z poniższych linków:

- test umiejętności współpracy - https://tinyurl.com/32rpw7wf
- test predyspozycji - https://tinyurl.com/2v6kneeh

Następnie zduplikuj je i umieść na swoim koncie Microsoft, wybierz przycisk "Udostępnij" i po wybraniu odpowiednich ustawień skopiuj adresy poszczególnych formularzy i udostępnij je uczniom do wypełnienia bezpośrednio przed rozpoczęciem testu.

4. Wydrukuj karty z kodami QR kierującymi do filmików dla uczniów (każdy kod QR prowadzi do jednej listy odtwarzania np. dotyczącej konkretnej lekcji lub .

5. Zastanów się, czy chcesz nagrodzić uczniów, których zespoły wykonały projekty zgodnie z założeniami, np. przyznając im ocenę celującą.

Zajęcia mogą być realizowane w oparciu o 3 przykładowe projekty:

1. Lokalizacje lotów samolotów: https://aviationstack.com (w filmach używamy tego projektu).

2. Weryfikacja numerów telefonów z podaniem danych o operatorze i kraju: https://numverify.com

3. Alert pogodowy dla 3-dniowej prognozy: https://wttr.in/

Każdy zespół może sobie wybrać dowolny inny projekt (inne API, inne funkcjonalności) - ważne, aby zachować układ: aplikacja, dokumentacja, strona internetowa o produkcie. Jeśli uznasz za stosowne, aby ułatwić uczniom zadanie, zasugeruj im wybranie projektu dot. lokalizacji lotów samolotów - będą wówczas odtwarzać przykłady zamieszczone w filmach instruktażowych. Z punktu widzenia celu zajęć ważne jest przeprowadzenie procesu budowania aplikacji, aby młodzież nauczyła się współpracy w zespole. Najbardziej ambitne zespoły mogą - za Twoją zgodą - dowolnie kształtować wykonywane przez siebie projekty - nie ograniczaj ich inwencji, nawet, jeśli masz poczucie, że mogą przewyższać Cię wiedzą techniczną. Szczególnie wtedy pozostaw uczniom sporą swobodę w działaniu, a zaskoczy Cię ich kreatywność i umiejętność rozwiązywania problemów.

Informacja dla szkolnego administratora sieci:

w celu unifikacji i ułatwienia pracy przygotowaliśmy kompletny system operacyjny Linux Ubuntu 20.04 LTS w wersji maszyny wirtualnej OVA dla środowiska Virtualbox. Przed rozpoczęciem zajęć na wszystkich komputerach należy zainstalować ** Virtualbox** oraz zaimportować do niego maszynę OVA - poniżej lista kroków do wykonania:

- ze strony https://www.virtualbox.org/wiki/Downloads należy pobrać i zainstalować Virtualbox w wersji 6 odpowiedniej dla używanego w szkole systemu operacyjnego, przykładowo: dla systemu Windows może to być https://download.virtualbox.org/virtualbox/6.1.32/VirtualBox-6.1.32-149290-Win.exe i zainstalować w systemie
- ze strony https://tinyurl.com/popo-ova pobrać plik linux_ubuntu.ova i zapisać na dysku (**UWAGA - plik ma ok. 7 GB!!**)
- w środowisku Virtualbox wykonać Import maszyny OVA
- po sprawdzeniu poprawności uruchamiania systemu można usunąć plik linux_ubuntu.ova

Faza realizacyjna:

1. Wprowadź uczniów w kontekst zajęć, tzn. opowiedz o tym, że:

- celem zajęć jest wspólna praca nad projektem, poznanie języka programowania, narzędzi edycyjnych etc.,
- będą pracować nad stworzeniem prostej aplikacji, ucząc się przy tym języka programowania Python oraz pozostałych narzędzi,
- będą pracować w zespołach, na których skład nie mają wpływu – odzwierciedla to prawdopodobne realia ich przyszłej pracy zawodowej,
- mają do dyspozycji wspomagające materiały wideo i Twoje wsparcie.
- rozdaj uczniom wydrukowane karty z kodami QR do list odtwarzania filmów.

Jeśli tak postanowiłeś, to jest dobry moment, aby zaproponować uczniom, że zwycięskie (lub wszystkie) zespoły, które ukończyły i oddały działające aplikacje wraz pozostałymi materiałami (strona internetowa etc.) otrzymają

oceny celujące. Zawsze proponuj nagrodę, aby motywować uczniów do podjęcia wyzwania.
2. Przeprowadź testy:

- predyspozycji i ról w zespole służący przygotowaniu podziału na zespoły (ok. 5 minut)
- umiejętności współpracy i komunikacji w zespole, który stanowił będzie punkt odniesienia dla oceny nabycia kompetencji społecznych (ok. 10-15 minut).

3. Przedstaw zasady pracy podczas zajęć, zaproponuj tzw. kontrakt w którym zawarte będą zasady formalne (więcej informacji dot. zasad pracy w Przewodniku dla nauczycieli) oraz zaproś uczniów do zgłoszenia swoich zapisów.

4. Jeśli jeszcze nie było ku temu okazji, zapytaj uczniów, jaka jest ich znajomość języka Python oraz innych zagadnień omawianych w trakcie zajęć. Wykorzystaj tę wiedzę podczas podziału uczniów na zespoły.

Uczniom, którzy chcieliby samodzielnie poznać podstawy języka, zasugeruj, aby skorzystali z bezpłatnych szkoleń dostępnych w portalu OSE:

Podstawowy: https://it-szkola.edu.pl/kkurs,kurs,216 wraz z kodami w serwisie GitHub: https://github.com/klubmlodegoprogramisty/python/tree/main/poziom_podstawowy

Średnio zaawansowany: https://it-szkola.edu.pl/kkurs,kurs,217 wraz z kodami w serwisie GitHub: https://github.com/klubmlodegoprogramisty/python/tree/main/poziom_sredniozaawansowany

lub wskaż inne źródła, które uznasz za wartościowe.

5. Przeprowadź *Test umiejętności współpracy i komunikacji* w zespole (ok 15 minut) i zarchiwizuj wyniki.

6. Przeprowadź *Test predyspozycji i ról* (ok. 5 minut) i zapisz wyniki. Posłużą Ci do dokonania podziału uczniów na zespoły.

Faza podsumowująca:

1. Poinformuj uczniów, w jakim terminie poznają skład zespołów. Termin powinien być jak najkrótszy, aby uczniowie mogli nawiązać kontakt przed lekcją nr 1.

2. Zachęć do:

przeczytania *Przewodnika dla uczniów* i obejrzenia filmów dotyczących komunikacji i pracy w zespole i przeczytania *Przewodnika dla uczniów*,

niezwłocznego kontaktu w ramach zespołu i wyboru tematu, nad którym zespół będzie pracował,

UWAGA: Zmiana wyboru projektu dla aplikacji może nastąpić najpóźniej po lekcji 3, kiedy w trakcie samodzielnej pracy uczniowie mogą sprawdzić działanie różnych API - jest to jedynie opcja - sugerujemy pozostanie przy pierwotnych wyborach.

- stworzenia wewnątrz-zespołowych kontraktów regulujących zasady współpracy (więcej informacji w Przewodniku dla nauczycieli) i nadania nazw swoim zespołom. Celem jest stworzenie poczucia przynależności i integracji zespołu.

Czynności po lekcji:

1. Dokonaj podziału na zespoły czteroosobowe. - jeśli to konieczne ze względu na arytmetykę, jeden z zespołów może liczyć od 2 do 5 osób (więcej informacji o zespołach i podziale w *Przewodniku dla nauczycieli*);

- w idealnej sytuacji każdy członek zespołu reprezentuje inny rodzaj osobowości/styl komunikacji (w teście – kolor);. jeśli nie jest to możliwe, to decyduje drugi w kolejności wybrany kolor;
- uwzględnij parytet płci;. idealnie jest, gdy w zespole są dwie dziewczyny i dwóch chłopców; jeśli nie ma takiej możliwości, to dążymy do jednorodności zespołu pod względem płci;
- uwzględnij poziom wiedzy i zaawansowania, jeśli chodzi o zagadnienia programistyczne lub znajomość pozostałych umiejętności merytorycznych wymaganych podczas zajęć (język HTML, wykorzystanie programów graficznych, tworzenie dokumentacji); zadbaj więc, aby w jednym

zespole, nie znalazły się wyłącznie osoby najbardziej lub najmniej zaawansowane;

- zadbaj, aby w jednym zespole nie znalazły się osoby silnie dominujące lub jawnie ze sobą skonfliktowane.

2. Przekaż informację o składach zespołów uczniom, optymalnie co najmniej na 2 dni przed lekcją 1.

Temat: Lekcja 1 - start, przygotowanie środowiska

Autor: Adam Jurkiewicz

Opis ogólny:

W trakcie tego spotkania skupiamy się na przygotowaniu środowiska pracy, a więc OVA w Virtualbox, venv w PyCharm. Dodatkowo młodzież pozna różnego rodzaju licencje na materiały i oprogramowanie, oraz co najważniejsze - stworzy swój pierwszy program w Pythonie - zgodnie z zasadą szybkich efektów pracy, aby nie zniechęcić się na początku.

Podstawowe treści omawiane w materiale:

- licencje na oprogramowanie
- PyCharm i prosty projekt programistyczny
- serwis GitHub - współpraca wielu osób nad projektem
- serwisy z darmowymi materiałami graficznymi
- Writer - edytor tekstów
- GIMP - edytor grafiki
- Bluefish - edytor HTML

Przebieg lekcji

Czynności przygotowawcze przed lekcją:

1. Przygotowanie do zajęć. Należy zweryfikować działanie VirtualBox na komputerach szkolnych - to krytyczne dla powodzenia projektu.

2. Warto zapoznać się z filmami przed pierwszymi zajęciami.

3. W szczególności zalecamy filmy **a02**, **a03**, **a04** - aby przetestować działanie aplikacji i być gotowym na ewentualne problemy uczniów, np. literówki

Faza wstępna:

1. Rozpoznanie wiedzy uczniów. Warto podyskutować chwilę o ich znajomości serwisu GitHub i o tym, jak wykorzystują je firmy do wspólnej pracy programistów.

Faza realizacyjna:

1. **Konfiguracja IDE PyCharm, tworzenie konta w GitHub.com**

Z uwagi na fakt, że w OVA w VirtualBox wszystko już jest zainstalowane, uczniowie tylko zakładają sobie konta na GitHub. Tę czynność w ogóle można zlecić do wykonania samodzielnie w domu przed zajęciami.

2. **Przygotowujemy środowisko venv dla lokalnego projektu**

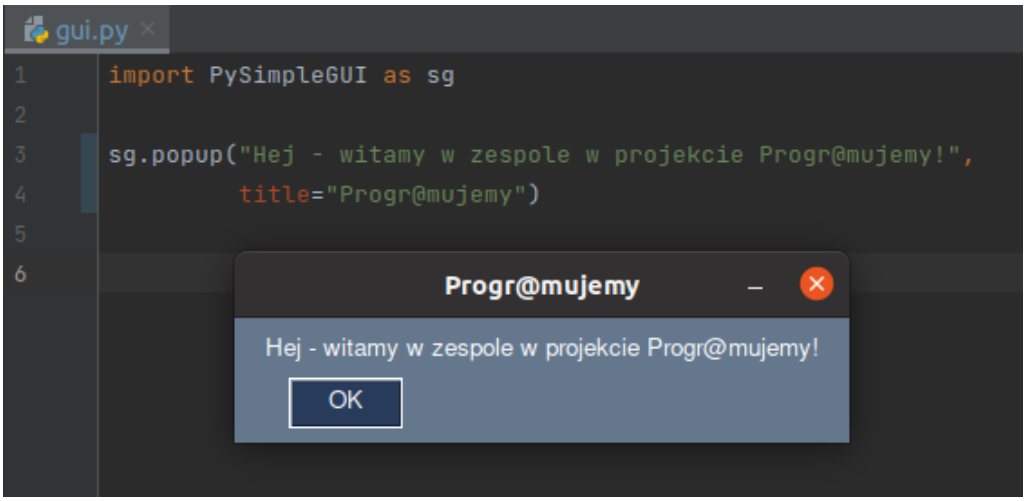
Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

3. **Plik requirements.txt - zewnętrzne moduły, własne pliki py w projekcie**

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Zwracamy uwagę na zewnętrzne moduły - można pokazać stronę https://pypi.org/ (to repozytorium zewnętrznych bibliotek do Pythona, np.: https://pypi.org/project/PyTechBrain/)

4. Minimalny program z wykorzystaniem PySimpleGUI

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Uczniowie testowo wykonują lokalne środowisko virtualenv w PyCharm i tam dla testów wykonują bardzo prostą aplikację:



```
import PySimpleGUI as sg

sg.popup(„Hej - witamy w zespole w projekcie Progr@mujemy!”,
         title=”Progr@mujemy”)
```

5. Skąd będziemy czerpać grafiki? Pixabay, Freepik i Flaticon.

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Możemy przypomnieć o licencjach, zgodnie z zapisami podstawy programowej uczniowie mają rozróżniać różne licencje.

6.Uruchamiamy edytor tekstów - format strony.

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Możemy przypomnieć o licencjach, zgodnie z zapisami podstawy programowej uczniowie mają rozróżniać różne licencje.

7. Uruchamiamy edytor grafiki - otwieramy przykładowy plik graficzny.

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Możemy przypomnieć o licencjach, zgodnie z zapisami podstawy programowej uczniowie mają rozróżniać różne licencje.

> **8. Uruchamiamy edytor HTML - podstawowa strona z szablonu.**
>
> Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Możemy przypomnieć o licencjach, zgodnie z zapisami podstawy programowej uczniowie mają rozróżniać różne licencje.
>
> **9. Edytor HTML: Bootstrap - https://getbootstrap.com/**
>
> Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Przypomnijmy, jak wiele oprogramowania jest tworzonego na otwartych licencjach, np.: https://github.com/twbs/bootstrap/blob/v5.0.2/LICENSE
>
> The MIT License (MIT)
>
> Copyright (c) 2011-2021 Twitter, Inc.
>
> Copyright (c) 2011-2021 The Bootstrap Authors
>
> Permission is hereby granted, free of charge, to any person obtaining

Faza podsumowująca:

1. Nauczyciel może podsumować wykonane zadanie, uczniowie mogą sprawdzić zawartość katalogu wirtualnego środowiska Python dla swojego projektu.

2. Do obejrzenia i samodzielnej pracy przeznaczone są w tym momencie następujące filmy:

1. VirtualBox w Windows i jak importować maszynę OVA - aby pracować niezależnie od szkoły

2. PySimpleGui - dokumentacja, przykłady użycia

3. Edytor tekstów: nagłówek i stopka

4. Edytor tekstów: style i spis treści

5. Edytor grafiki: zmiana rozmiaru i zapis XCF

6. Edytor HTML: różne znaczniki meta

Cele operacyjne (językiem ucznia):

- Poznasz różne aplikacje
- Powtórzysz wiedzę o licencjach na materiały i oprogramowanie

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Python 3 (lub nowszej wersji), dostępna pod adresem: https://docs.python.org/.
- Dokumentacja wyboru licencji: https://creativecommons.pl/wybierz-licencje/

Wskazówki metodyczne:

- można wskazać na różne rodzaje licencji różnych systemów operacyjnych i samego Pythona w kontekście prawa autorskiego.

Temat: Lekcja 2 - podstawowe typy danych i konstrukcje programistyczne w Python

Autor: Adam Jurkiewicz

Opis ogólny:

W trakcie tego spotkania uczniowie poznają podstawowe typy danych , konstrukcje pętli oraz instrukcji warunkowych. Dla aplikacji ogólnych wprowadzamy listy, warstwy i kontenery. Te elementy są ściśle powiązane z podstawą programową.

Podstawowe treści omawiane w materiale:

- podstawowe typy danych w Python
- kolekcje: listy i słowniki
- importowanie funkcji z zewnętrznych modułów
- pętla iteracyjna `for...in...` oraz instrukcja warunkowa `if...else...`
- listy w edytorze tekstów
- warstwy w edytorze graficznym
- kontenery w edytorze HTML

Przebieg lekcji

Czynności przygotowawcze przed lekcją:

1. Sprawdzamy, czy każda osoba ma działające oprogramowanie VirtualBox, dostęp do internetu. Możemy samodzielnie spróbować uruchomić kody, przeanalizować ich działanie, skonfrontować naszą wiedzę z filmami instruktażowymi.

Faza wstępna:

1. Możemy dopytać się uczniów, czy wykonali zadania przeznaczone do pracy samodzielnej, czy udało im się uruchomić VirtualBox i zaimportować maszynę `OVA`

Faza realizacyjna:

1. Podstawowe typy danych w Python, zmienne

W języku Python istnieją zmienne — nie jest to bardzo odkrywcze stwierdzenie. Zmienne zawierają wartości różnego typu, i to na podstawie analizy tych wartości interpreter Pythona określa typ zmiennej. Nie deklarujemy typu zmiennej, musimy pamiętać, że typ zmiennej zmienia się z jej zawartością. Musimy również pamiętać, że w języku Python rozróżniamy wielkość liter: zmienna linux to coś innego niż `Linux`, a funkcja `Print()` to nie jest `print()`. To dosyć prosty materiał — wystarczy pokazać film i postępować zgodnie ze wskazówkami. Jednak uwaga! Pośrodku filmu ujawnia się błąd! To jest specjalne — aby pokazać, w jaki sposób Python komunikuje problemy i błędy w kodzie. W serwisie GitHub pozostawiam kod z błędem, który uczniowie muszą samodzielnie poprawić. Wprowadzam tam od razu konwencję F-String, która jest obecnie (rok 2021/2022) najpopularniejsza w Pythonie.

Przykład krótkiego wywołania F-String (pamiętamy o literze `f` przed znakami cudzysłowów):

```
some_value = 3.1415

some_text = f”PI Value is: {some_value}”
print(some_text)

# inny sposób:
print(f”PI Value is: {some_value}”)
```

2. Typy zaawansowane: listy, słowniki

W języku Python spotkamy się niejednokrotnie z sekwencjami - to obiekty, które składają się z wielu innych obiektów dowolnego typu. Na początku to może być bardzo skomplikowane zdanie. W praktyce najczęściej mówimy o prostych sytuacjach. Tu opisane są struktury danych, które mogą przechowywać obiekty różnych typów danych.

- `<class ‚list’>` Struktura **listy** (uporządkowanej kolekcji danych, zachowujących kolejność wprowadzania); np. `[3, 4, 5, 5, 6, 2, 4, 1]` - tutaj przykładowa lista zawierająca oceny uczennicy Beata

- <class ‚dict'> Struktura **słownika** (nieuporządkowanej kolekcji elementów składających się z klucza i wartości); np. {1: „Adam", 2: „Jurkiewicz", 3: „Linux"} - tutaj przykładowy słownik z różnymi kluczami i wartościami Tak naprawdę od wersji 3.6 Pythona słowniki zachowują pewne uporządkowanie, lecz traktujmy je jako nieuporządkowane, a więc niezachowujące kolejności występowania elementów. W przypadku słowników zwróćmy uwagę na znaki dwukropka rozdzielające klucz i wartość.

3. Importowanie z zewnętrznych modułów

Musimy teraz powiedzieć sobie o tych różnych możliwościach importu oraz o konsekwencjach. Oczywiście nie podamy tu wszystkich, tylko te, które będą nam niezbędne:

- import modul- w ten sposób wczytujemy cały moduł, a do jego poszczególnych elementów odwołujemy się poprzez zapis: modul.funkcja()
- from modul import *- w ten sposób wczytujemy wszystkie elementy z modułu, a odwołujemy się poprzez zapis funkcja() , bez podawania jawnie nazwy modułu
- from modul import funkcja_a - w ten sposób wczytujemy tylko pewną konkretną funkcję, a odwołujemy się poprzez zapis funkcja_a(), bez podawania jawnie nazwy modułu
- from modul import funkcja_a as funkcja_b - w ten sposób wczytujemy tylko pewną konkretną funkcję i nadajemy jej własną nazwę, a odwołujemy się poprzez zapis funkcja_b(), bez podawania jawnie nazwy modułu

Pierwszy sposób jest najogólniejszy — stosujemy go wtedy, kiedy mamy pewność, że chcemy odwołać się do pewnych szczególnych elementów (jak np. sys.path).

Drugi sposób pozwala nam w łatwy sposób mieć szybki dostęp (nie musimy dużo pisać) do wszystkich funkcji w danym module. To dobry sposób, jeśli nie korzystamy z wielu modułów ani nie mają one dużo kodu.

Trzeci sposób jest lepszy od drugiego, gdyż wczytujemy tylko te funkcje, które potrzebujemy w naszym programie. Dzięki temu unikniemy wczytania dwóch różnych funkcji o tej samej nazwie.

Czwarty sposób stosujemy, kiedy nazwa funkcji jest długa i chcemy sobie zaoszczędzić czasu pisania lub trafiamy na dwie różne funkcje z dwóch modułów, lecz o tej samej nazwie.

Najczęściej stosujemy zapis trzeci - from modul import funkcja_a.

Tu warto zauważyć, że na filmie jest już utworzony plik i kawałek kodu skopiowany z serwisu GitHub. Zrobiłem to, aby zaoszczędzić czasu w filmie na tłumaczenie, gdyż jest to dosyć trudny temat — proponuję to przećwiczyć przed przystąpieniem do zajęć, a uczniowie niech samodzielnie w domu też to przećwiczą, poza ćwiczeniami w szkole.

4.Pętla for i listy

Pozwala ona wykonywać blok kodu określoną liczbę razy. Python udostępnia dodatkowo dwie instrukcje, które odpowiadają za kontrolę działania bloku kodu wewnątrz pętli:

- break – kończy działanie pętli, a Python przechodzi do dalszej części instrukcji – następujących po bloku pętli
- continue – kończy iterację bieżącej pętli, a Python wraca do początku pętli, wyrażenie warunkowe jest ponownie sprawdzane, aby określić, czy pętla zostanie wykonana ponownie, czy zakończyć i przejść dalej
- Tutaj na filmach w PyCharm są utworzone 2 pliki, podczas gdy w GitHub jest to jeden plik. To dla przejrzystości filmów. W szkole można wykonać to dowolnie — albo dwa pliki jak na filmie, albo jeden duży jak w GitHub. Pozostawiam to decyzji nauczyciela.

Instrukcja warunkowa if ... else ...

Badamy w nich warunek, w zależności od wyniku tego badania wykonujemy pewien blok kodu, lub wykonujemy inny, a być może nie wykonujemy żadnego. Bardzo uważnie musimy dobierać warunki logiczne. Możemy sprawdzić, czy wartość obiektu jest mniejsza, większa, równa pewnej stałej wartości. Możemy sprawdzić, czy dwa obiekty na ekranie są w pewnej odległości od siebie. W języku Python używamy następującej składni (zapisu):

```
if warunek_1:
    # blok kodu gdy warunek_1 jest prawdą
elif warunek_2:
    # blok kodu gdy warunek_2 jest prawdą
elif warunek_3:
    # blok kodu gdy warunek_3 jest prawdą
elif warunek_n:
    # blok kodu gdy warunek_n jest prawdą
else:
    # blok kodu wykonywany wówczas, kiedy żaden z powyższych
warunków nie jest spełniony
```

Ważne!

Pamiętamy o blokach kodu — wcięcie 4 spacje!

Minimalnie musimy podać jeden warunek, a ich maksymalna ilość jest nieograniczona. Warunki są sprawdzane w kolejności, w jakiej je zapisujemy. W momencie, kiedy Python „napotka" warunek, który jest prawdziwy, przerywa sprawdzanie. Tu pokazujemy film i postępujemy zgodnie ze wskazówkami zawartymi na filmie...

W drugiej części filmu (pobieranie nagłówków) zwracajmy uwagę na odwołania do elementów listy poprzez indeks (zaczynamy liczyć od 0). W tym momencie pokazujemy też kolejny typ złożonych danych - Tupla to kolekcja bardzo podobna do listy. Główna różnica - Tupla nie jest modyfikowalna.

6.Edytor tekstów: listy numerowane i nienumerowane

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Pamiętajmy, aby zapisać plik na końcu ;-)

7. Edytor grafiki: warstwy i dodanie elementu

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Ikona bank.png jest zawarta w repozytorium (Uicons z serwisu https://www.flaticon.com/uicons) Proszę zwrócić uwagę, że na filmie mówimy o warstwach - ich znaczniki warto wskazać jawnie (po prawej stronie okna GIMP'a).

8. Edytor HTML: containers

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Zwracamy uwagę na strukturę ... , która określa zasięg działania parametru:

```
<div class=”container-lg”>
    <h1>This is our HTML Page with bootstrap</h1>
</div>
```

Faza podsumowująca:

1. Nauczyciel może podsumować wykonane zadania, wskazując na nowe, ważne elementy:

Z zakresu programowania: wcięcia 4 spacje i bloki kodu

Z zakresu HTML: składnię

Do obejrzenia

i samodzielnej pracy przeznaczone są w tym momencie następujące filmy:

1. Wyświetlamy informację.

2. Rozpakowywanie tupli — pythonizm.

3. Pętla while True - sterowanie programem PySimpleGUI.

4. Dodajemy elementy przycisków.

5. Dodajemy wyświetlanie obrazków.

6. Poznajemy sposoby wprowadzania danych.

7. Poznajemy sterowanie.

8. PySimpleGui — tworzymy prosty program okienkowy.

9. Wyświetlanie większej ilości danych.

10. Edytor tekstów: zrzut zawartości okna aplikacji i dodanie do tekstu.

11. Edytor grafiki: warstwy i dodanie tekstu.

12. Edytor HTML: różne elementy na stronie (display, images, lists).

UWAGA: Warto zaznaczyć, aby uczniowie dokładnie zapoznali się z tymi filmami, zwłaszcza ci, którzy są odpowiedzialni za kwestie programistyczne. To spora dawka materiału, która opisuje elementy sterujące dla aplikacji okienkowej tworzonej za pomocą biblioteki PySimpleGUI.

Cele operacyjne (językiem ucznia):

- Poznasz podstawowe typy danych w języku Python.
- Powtórzysz wiedzę o różnych elementach w edytorach.

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Python 3 (lub nowszej wersji), dostępna pod adresem: https://docs.python.org/.
- materiały kursu OSE:s://github.com/klubmlodegoprogramisty/python/tree/main/poziom_podstawowy
- materiały książki: https://github.com/abixadamj/helion-python/tree/main/Rozdzial_1

Wskazówki metodyczne:

Przykłady ćwiczeń w pliku 99_excersises.py.

- Jako ćwiczenie można zadać napisanie programu, który bazując na roku urodzenia obliczy, w którym roku dana osoba będzie mieć 100 lat.

```
actual_year = 2022
birth = 1974
my_name = „Adam Jurkiewicz”

print(f”Hi {my_name}, we’ll try to compute year of your 100 ;-)”)
age = actual_year - birth
when_100 = birth + 100
text_when = f”Dear {my_name}, you will be 100 years old at
{when_100}. It will be {100 - age} years from now.”
print(text_when)
```

- Jako ćwiczenie wykorzystujące nowe elementy proponuję: napisz program, który sprawdzi, czy elementem kolekcji jest książka i poda jej tytuł. Mamy dwie kolekcje:

1. Lista zawierająca tuple [o niej samej będzie jeszcze] (rodzaj, tytuł) : [(„CD”, „Album 1”), („DVD”, „Album 3”), („Book”, „Moby Dick”), („MP3”, „Collection 2”)]

2. Słownik o strukturze: {„CD”: „Album 1, „MP3”: „Collection 2”, „DVD”: „Album3”, „Book”: „Moby Dick”}

```
my_list = [(„CD”, „Album 1”), („DVD”, „Album 3”), („Book”, „Moby Dick (list)”),(„MP3”, „Collection 2”)]
my_dict = {„CD”: „Album 1”, „MP3”: „Collection 2”, „DVD”: „Album 3”, „Book”:„Moby Dick (dict)”}

for element in my_list:
    if element[0] == „Book”:
        print(f”I found a book in a list: {element[1]}”)

for element in my_dict:
    if element == „Book”:
        print(f”I found a book in a dict: {my_dict[element]}”)
```

Temat: Lekcja 3 - requests, API, słowniki i JSON

Autor: Adam Jurkiewicz

Opis ogólny:

W trakcie tego spotkania poznajemy pierwszy sposób wykorzystania danych z zewnętrznego API (serwisu podającego dane). Zapoznamy się ze słownikiem i formatem JSON.

Podstawowe treści omawiane w materiale:

- PythonConsole w PyCharm
- format `JSON` a słowniki w Pythonie
- wykorzystanie pętli `for` do wyświetlania wszystkich elementów słownika
- struktury list i słowniki jako wartości słowników
- formaty plików, elementy HTML

Przebieg lekcji

Czynności przygotowawcze przed lekcją:

1. Sprawdzamy, czy każda osoba ma działające oprogramowanie VirtualBox, dostęp do internetu. Możemy samodzielnie spróbować uruchomić kody, przeanalizować ich działanie, skonfrontować naszą wiedzę z filmami instruktażowymi.

Faza wstępna:

1. Możemy dopytać się uczniów, czy wykonali zadania przeznaczone do pracy samodzielnej, czy mają jakiekolwiek swoje kody źródłowe.

Faza realizacyjna:

1. Poznajemy `Python Console` w PyCharm + Wykorzystujemy `requirements.txt` i instalujemy niezbędne elementy: `requests`

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

2. Wykonujemy request z serwisu https://fastapi.jurkiewicz.tech/ i pokazujemy odczytane dane

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

Możemy zwrócić uwagę na złożoność obiektu zwracanego funkcją `get` :

```
import requests
# wykonujemy połączenie typu GET - analogicznie jak przeglądarka WWW
https_request = requests.get(„https://fastapi.jurkiewicz.tech/”)

# wyświetlamy zawartość
print(https_request.text)


# pewien sposób stworzenia obiektu JSON
print(https_request.json())
```

3. `JSON` i słowniki w Python

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

Tu również możemy zwrócić uwagę na złożoność obiektu zwracanego funckcją get jak w

poprzednim przykładzie.

4. Użycie pętli `for` dla pokazania elementów słownika z serwisu https://fastapi.jurkiewicz.tech/

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

5. Listy jako elementy słowników

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

6. Słowniki jako elementy słowników

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

7. Edytor tekstów: eksport dokumentu do formatu PDF

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

8. Edytor grafiki: eksport obrazu jako PNG

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

9. Edytor HTML: różne elementy na stronie (address, listy, user input, sample output)

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

Faza podsumowująca:

1. Nauczyciel może podsumować wykonane zadanie, uczniowie mogą sprawdzić zawartość

katalogu wirtualnego środowiska Python dla swojego projektu.

2. Do obejrzenia i samodzielnej pracy przeznaczone są w tym momencie następujące filmy:

1. Sprawdzamy dokumentację dla przykładowych API:

2. https://aviationstack.com/documentation

3. https://numverify.com/documentation

4. https://wttr.in/:help

3. Generowanie API_KEY dla wybranego projektu (Aviationstack) - http

s://aviationstack.com/signup/free

4. Poznajemy kody odpowiedzi API: poprawnych i błędnych (3 przykłady

dla każdego projektu)

5. Tworzymy własne repozytorium, pamiętamy o `.gitignore`,

`README.md` oraz licencji

UWAGA: Warto zaznaczyć, aby uczniowie dokładnie zapoznali się z tymi filmami, zwłaszcza ci, którzy są odpowiedzialni za kwestie programistyczne. W tym momencie mogą już realnie zabrać się za tworzenie aplikacji - mają 75% informacji niezbędnych do jej wytworzenia. To spora dawka materiału, zwłaszcza warto zapoznać się z dokumentacjami projektu, który dana grupa chce realizować; w przykładach są tylko 3, a być może grupa wybierze zupełnie inne API.

Cele operacyjne (językiem ucznia):

- poznasz PythonConsole w PyCharm
- poznasz format JSON oraz słowniki w Pythonie
- wykorzystasz pętlę for do wyświetlania wszystkich elementów słownika
- zaznajomisz się ze strukturą list i słowników jako wartości słowników
- zobaczysz formaty plików, elementy HTML

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Python 3 (lub nowszej wersji), dostępna pod adresem: https://docs.python.org/.
- materiały kursu OSE: https://github.com/klubmlodegoprogramisty/python/tree/main/poziom_podstawowy
- materiały książki: https://github.com/abixadamj/helion-python/tree/main/Rozdzial_1

Wskazówki metodyczne:

Jako ćwiczenie wykorzystujące nowe elementy proponuję: napisz program, który wyświetli

różne informacje z API.

Przykład w pliku 99_excersises.py

```
import requests


https_request = requests.get(„https://fastapi.jurkiewicz.tech/”)
api_data = https_request.json()
# teraz wyciągamy konkretny element z naszego słownika
header = „Client’s headers”
print(f”IP klienta to {api_data[header][‚x-forwarded-for’]}”)
```

Temat: Lekcja 4 - funkcje, różne interfejsy API, Commit/Push do repozytorium

Autor: Adam Jurkiewicz

Opis ogólny:

W trakcie tego spotkania poznajemy sposoby definiowania funkcji i przestrzenie w Python (ang. namespace); zapoznamy się ze skryptem odczytującym i prezentującym dane z przykładowego API. W drugiej części pracujemy ze zdalnym repozytorium GitHub.

Podstawowe treści omawiane w materiale:

- definiowanie funkcji
- koncepcja przestrzeni nazw w Python
- praca ze zdalnym repozytorium GitHub

Przebieg lekcji

Czynności przygotowawcze przed lekcją:

1. Sprawdzamy, czy każda osoba ma działające oprogramowanie VirtualBox, dostęp do internetu. Każdy powinien mieć konto w serwisie GitHub, pamiętać swój login i hasło. Możemy samodzielnie spróbować uruchomić kody, przeanalizować ich działanie, skonfrontować naszą wiedzę z filmami instruktażowymi.

Faza wstępna:

1. Możemy dopytać się uczniów, czy wykonali zadania przeznaczone do pracy samodzielnej, czy mają jakiekolwiek swoje kody źródłowe.

Faza realizacyjna:

1. Definiowanie funkcji w Python.

Funkcja - to zdefiniowany blok kodu, który realizuje pewne zadania, może bazować na

różnych danych wejściowych, zwanych parametrami. Funkcja może „zwrócić” nam pewne wartości.

Ogólny sposób definiowania funkcji w Pythonie wygląda tak:

```
def nazwa_funkcji(parametr_1, parametr_2):
    # blok kodu
    return zwracany_obiekt
```

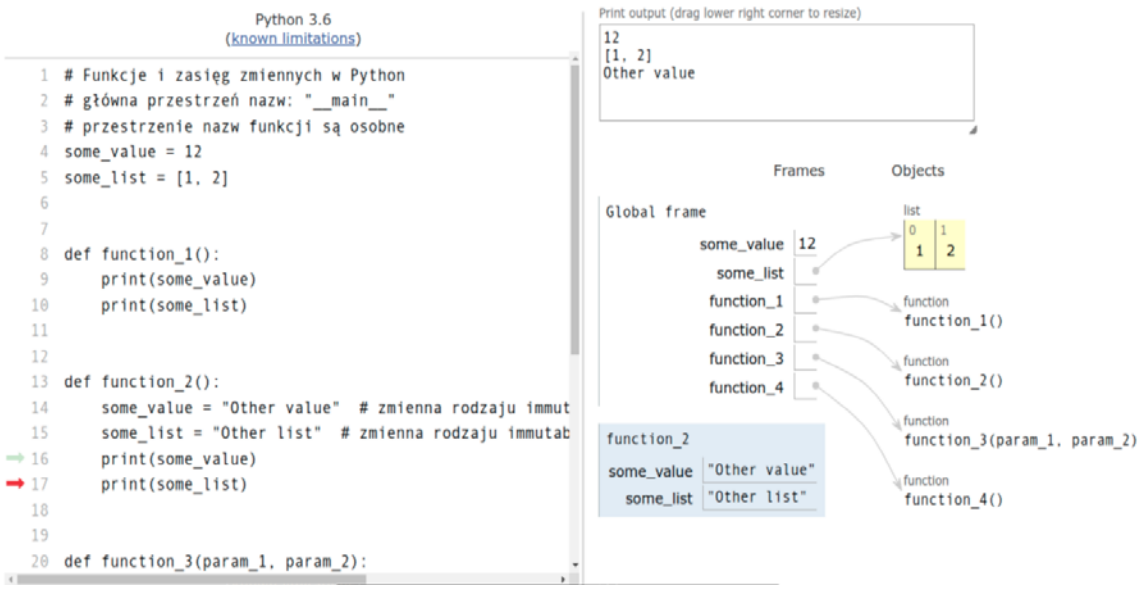
Zwróćmy uwagę na pewne ważne elementy:

- słowo kluczowe `def` (ang. define) - mówi nam, że to będzie definicja funkcji
- nazwa naszej funkcji zapisana jest małymi literami, jeśli zawiera dwa lub więcej słów oddzielamy je znakiem podkreślenia (to zasady z dokumentu PEP 8)
- parametry umieszczone są w nawiasie i rozdzielone przecinkami; wywołując funkcję musimy umieścić tyle samo argumentów, co parametrów w definicji (Uwaga! To nie jest do końca prawda, ale na tą chwilę przejmiemy takie założenie, później, przy tłumaczeniu konkretnego kodu, poznamy więcej możliwości.)
- znak dwukropka oznacza, że następnie występuje blok kodu i muszą być zachowane
- słowo kluczowe `return` (pol. zwróć/oddaj) - mówi nam, że to koniec działania funkcji i `zwraca zwracany_obiekt`

2. Funkcje i zasięg zmiennych w Python.

W języku Python (a także w innych) istnieje pojęcie „przestrzeni nazw” - w ich obrębie każda nazwa obiektu musi być niepowtarzalna. Przestrzeń nazw najczęściej związana jest z funkcją, modułem lub plikiem, a także z obiektem.

Poniżej widok serwisu https://pythontutor.com/, w którym możemy wizualizować kod (https://tinyurl.com/popo-namespace)



Przykładowy kod pokazuje przestrzenie nazw i widoczność zmiennych w pliku 02_namespace.py . Zwróćmy uwagę na kod - widzimy w nim, jak zmienne tworzone wewnątrz definicji funkcji *przesłaniają* sobą tak samo nazywające się zmienne, lecz

zdefiniowane w innej przestrzeni nazw.

```
# Funkcje i zasięg zmiennych w Python
# https://tinyurl.com/popo-namespace
# główna przestrzeń nazw: „__main__”
# przestrzenie nazw funkcji są osobne
some_value = 12
some_list = [1, 2]
def function_2():
    some_value = „Other value” # zmienna rodzaju immutable
    some_list = „Other list” # zmienna rodzaju immutable, bo
przypisanie innej wartości
    print(some_value)
    print(some_list)
def function_4():
    some_list.append(3) # zmienna rodzaju mutable, do listy dodajemy
element
print(some_value)
print(some_list)
```

3. Testujemy dostęp do danych API (3 przykłady dla każdego projektu)

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami. Jednak zależy zwrócić uwagę na zmienną api_key , która zawiera przykładowy klucz dostępu - na 99,999% będzie on nieaktywny w momencie testowania, więc uzniowie muszą użyć swoich kluczy, które generują samodzielnie w pracy własnej.

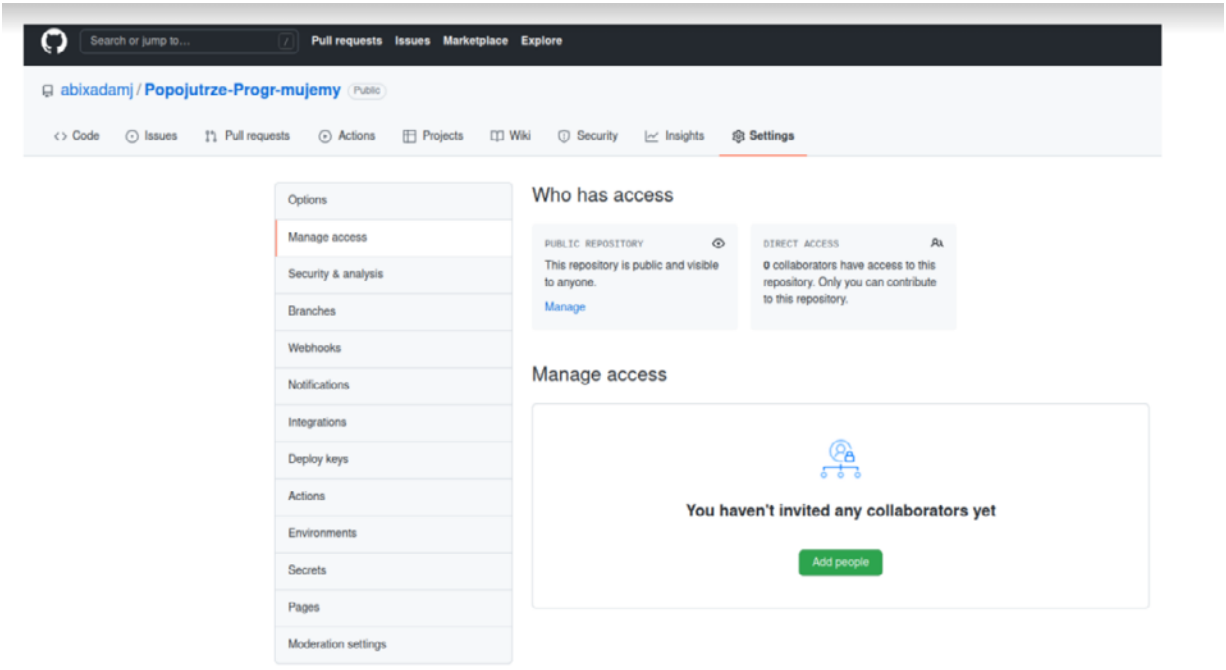
4. Replikacja projektu z GitHub do PyCharm (open via VCS) i dodanie lokalnego venv (Add interpreter) i w requirements.txt Install all packages a38

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

5. Dodanie do repozytorium pracy z aplikacją PySimpleGUI i Commit/Push a40

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

Na koniec lekcji tworzą konta w serwisie GitHub (https://github.com/), a osoba odpowiedzialna za programowanie tworzy repozytorium projektu i udziela innym dostępu do tego repozytorium *tu się zmieni zdjęcie..*



Faza podsumowująca:

1. Nauczyciel może podsumować wykonane zadanie, uczniowie mogą sprawdzić zawartość katalogu wirtualnego środowiska Python dla swojego projektu.

2. Należy dokładnie zaznaczyć sposób definiowania funkcji w Python - to ważny element.

3. Do obejrzenia i samodzielnej pracy przeznaczone są w tym momencie następujące filmy:

1. Sprawdzenie działania — skrypt odczytujący i prezentujący wybrane dane

2. Wysłanie projektu do serwisu GitHub

3. Przygotowanie dokumentacji i strony w HTML

Warto zaznaczyć, że to przedostatnia dawka teorii - w tym momencie już powinni mieć zręby aplikacji gotowe.

Cele operacyjne (językiem ucznia):

- poznasz PythonConsole w PyCharm
- poznasz format `JSON` oraz słowniki w Pythonie
- wykorzystasz pętlę `for` do wyświetlania wszystkich elementów słownika
- zaznajomisz się ze strukturą list i słowników jako wartości słowników
- zobaczysz formaty plików, elementy HTML

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Python 3 (lub nowszej wersji), dostępna pod adresem: https://docs.python.org/.
- materiały kursu OSE: https://github.com/klubmlodegoprogramisty/python/tree/main/poziom_podstawowy
- materiały książki: https://github.com/abixadamj/helion-python/tree/main/Rozdzial_1

Wskazówki metodyczne:

- Swoją pracę mogą wysłać do serwisu GitHub - należy ich zachęcić do tego, aby po obejrzeniu samodzielnie filmu spróbowali swoje skrypty tam wysłać.
- Jako ćwiczenie wykorzystujące nowe elementy proponuję: napisz program, który wyświetli różne informacje z API.

Przykład w pliku `99_excersises.py` :

```
import requests

https_request = requests.get(„https://fastapi.jurkiewicz.tech/”)
api_data = https_request.json()
# teraz wyciągamy konkretny element z naszego słownika
header = „Client’s headers”
print(f”IP klienta to {api_data[header][‚x-forwarded-for’]}”)
```

Temat: Lekcja 5 - praca z kluczami i wartościami słowników

Autor: Adam Jurkiewicz

Opis ogólny:

W trakcie tego spotkania poznajemy sposoby manipulowania wartościami kluczy słowników. Ta wiedza może być bardzo przydatna.

Podstawowe treści omawiane w materiale:

- aktualizowanie wartości kluczy słowników
- dodawanie nowych kluczy do słowników

Przebieg lekcji

Czynności przygotowawcze przed lekcją:

1. Sprawdzamy, czy każda osoba ma działające oprogramowanie VirtualBox, dostęp do internetu. Każdy powinien mieć konto w serwisie GitHub, pamiętać swój login i hasło. Możemy samodzielnie spróbować uruchomić kody, przeanalizować ich działanie, skonfrontować naszą wiedzę z filmami instruktażowymi.

Faza wstępna:

1. Możemy dopytać się uczniów, czy wykonali zadania przeznaczone do pracy samodzielnej, czy mają jakiekolwiek swoje kody źródłowe zaktualizowane w repozytorium w serwisie GitHub.

Faza realizacyjna:

1. Aktualizowanie wartości dla kluczy słowników.

Warto tu zwrócić uwagę na możliwe błędy podczas odwołania się do nieistniejącego klucza, jeśli chcemy zobaczyć wartość:

```
days = {
    0: {„min_temp”: 0, „max_temp”: 0, „warning”: False},
    1: {„min_temp”: 0, „max_temp”: 0, „warning”: False},
    2: {„min_temp”: 0, „max_temp”: 0, „warning”: False},
}
print(days[1][„min_temp”])
print(days[2][„max-temp”]) # tu wystąpi błąd
# Traceback (most recent call last):
# File „/usr/lib/python3.8/code.py”, line 90, in runcode
# exec(code, self.locals)
# File „<input>”, line 7, in <module>
# KeyError: ‚max-temp’

# poniżej aktualizujemy wartości już istniejących kluczy
days[1][„min_temp”] = 5
days[2][„max_temp”] = 15
```

2. Tworzenie słowników i dodawania do nich elementów.

Tu wystarczy pokazać film i postępować zgodnie ze wskazówkami.

Faza podsumowująca:

1. Nauczyciel może podsumować wykonane zadanie, można czas w tej lekcji wykorzystać na sesję pytań/odpowiedzi, na wspólną weryfikację dotychczasowej pracy, wiedzy. Warto zaznaczyć, że to ostatnia dawka teorii - w tym momencie już powinni mieć około 80% aplikacji i dokumentacji wykonane.

Cele operacyjne (językiem ucznia):

- poznasz metody aktualizacji danych w słownikach ćwiczenia praktyczne.

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka Python 3 (lub nowszej wersji), dostępna pod adresem: https://docs.python.org/.
- materiały kursu OSE: https://github.com/klubmlodegoprogramisty/python/tree/main/poziom
- materiały książki: https://github.com/abixadamj/helion-python/tree/

main/Rozdzial_1

- Wskazówki metodyczne: ta lekcja jest w większości już czasem dla uczniów na ostatnie zapytania.

Temat: lekcja 6 – podsumowanie projektu.

Autorzy: Konrad Kosieradzki, Adam Jurkiewicz, Rafał Kamiński

Opis ogólny:

W trakcie tego spotkania uczniowie otrzymają okazję do prezentacji wykonanych projektów oraz otrzymają informację zwrotną nt. swojej pracy zespołowej i jej efektów, zarówno od nauczyciela, jak i pozostałych uczniów. Na koniec uczniowie wykonają raz jeszcze test (kwestionariusz) kompetencji komunikacji i współpracy w zespole oraz dokonają oceny zajęć m.in. pod kątem ich atrakcyjności i przydatności.

Podstawowe treści omawiane w materiale:

Nie dotyczy - lekcja ma charakter podsumowująco-ewaluacyjny.

Przebieg lekcji

Czynności przygotowawcze przed lekcją:

1. Dokonaj oceny końcowej prac poszczególnych zespołów: .oceń, czy dany zespół uczniowski zrealizował swój projekt, tj. osiągnął zamierzony cel, przygotowana prosta aplikacja działa poprawnie (zgodnie z intencją zespołu), a kod jest poprawny i dobrej jakości. Zmianę w zakresie samooceny kompetencji komunikacji i współpracy w zespole zbadasz przy pomocy naszego testu (w trakcie zajęć).

Oceny stopnia osiągnięcia celów nie należy mylić z ocenami szkolnymi - przed lekcją 0 nauczyciel powinien zdecydować, czy za wykonane zadania będą przyznawane oceny (rekomendujemy motywowanie przez nagrodę, np. ocenę celującą za ukończenie projektu, tj. osiągnięcie opisanego powyżej celu zajęć w dwóch wymiarach).

Przy komentowaniu i sprawdzaniu prac możemy pomocniczo posłużyć się formatterem kodu Black (https://black.readthedocs.io/en/stable/) i sprawdzić, ile zmian wykona. Warto też pokazać uczniom to narzędzie i powiedzieć kilka słów nt. jego funkcjonalności i zastosowania.

2. Zapewnij możliwość prezentacji prac uczniów np. przy użyciu projektora, komputera z zainstalowanym środowiskiem PyCharm etc.

3. Zdecyduj, ile prac będzie mogło zostać zaprezentowanych (ideałem byłoby zaprezentować wszystkie).

4. Przygotuj *Test umiejętności* współpracy i komunikacji w zespole oraz Ankietę oceny innowacji.

Faza realizacyjna:

Zaproś uczniów do prezentacji prac zespołowych

1. Po zakończeniu prezentacji zachęć uczniów do komentarzy i moderuj dyskusję.

2. Podsumuj zajęcia zarówno w obszarze informatycznym jak i pracy zespołowej.

3. Przeprowadź *Test umiejętności współpracy i komunikacji* w zespole (ok 15 minut) i zarchiwizuj wyniki.

Przeprowadź *Ankietę oceny innowacji* (ok. 10 minut) i zarchiwizuj wyniki.

Faza podsumowująca:

1. Dokonaj analizy zmiany poziomu samooceny kompetencji komunikacji i współpracy w zespole, porównując wyniki testu wstępnego i końcowego. Skomentuj wyniki w obecności uczniów (podając pewne uogólnienia i statystyki, a nie wyniki poszczególnych uczniów).

2. Podsumuj zajęcia "Progr@muj w zespole" i podziękuj uczniom za zaangażowanie i wykonaną pracę. Jeśli to możliwe, zaproponuj uczniom jakąś formę wyróżnienia, np. możliwość upublicznienia (za ich zgodą) wyników prac na forum szkoły (w postaci aplikacji i stron www).

Przewodnik dla nauczycieli

Przewodnik dla nauczycieli informatyki dotyczący rozwoju kompetencji społecznych u uczniów

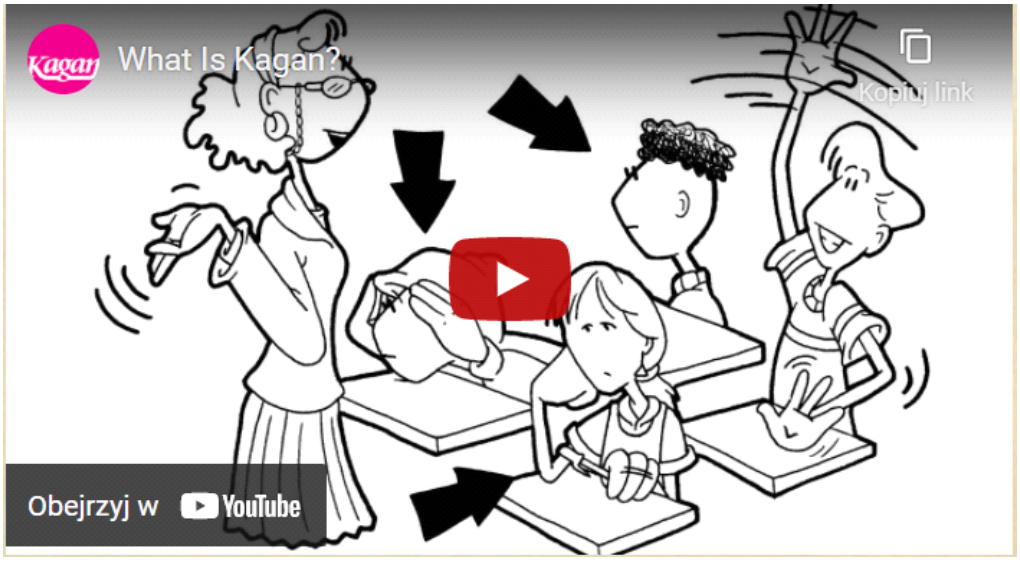
Zamiast wstępu

Droga Nauczycielko, Drogi Nauczycielu,

Treść tego przewodnika ma być wsparciem podczas przygotowania i prowadzenia zajęć w ramach projektu „Progr@muj w zespole". Dlatego też w wielu miejscach nawiązuje do zaproponowanego scenariusza zajęć i materiałów przygotowanych dla uczniów.

Idea naszego projektu jest w dużej mierze oparta na metodzie *Cooperative Learning* rozwijanej m.in. przez Davida i Rogera Johnsonów[1] oraz Spencera Kagana[2]. W ich publikacjach można znaleźć szczegółowe opisy metody, wyniki badań naukowych i bogatą bibliografię. Metoda jest również opisana w polskim piśmiennictwie[3].

Tu, w przewodniku, ograniczymy się do podstawowych założeń i informacji. Dla preferujących przekaz wizualny proponujemy, jako wprowadzenie, krótki filmik od Spencera Kagana:



1 http://www.co-operation.org, dostęp 06.01.2022
2 https://www.kaganonline.com, dostęp 06.01.2022
3 Stanisław Bobula, Norbert Karaszewski, Jakub Kołodziejczyk, Katarzyna Salamon-Bobińska Sesja II/9, Nauczanie kooperatywne (uczenie się we współpracy), SEO

Uczenie się we współpracy (*Cooperative Learning*[4]) postrzega ucznia jako osobę myślącą, zdolną do tworzenia własnych teorii i weryfikowania ich.

Według Spencera Kagana metoda ta opiera się na czterech podstawowych elementach wspólnej nauki:

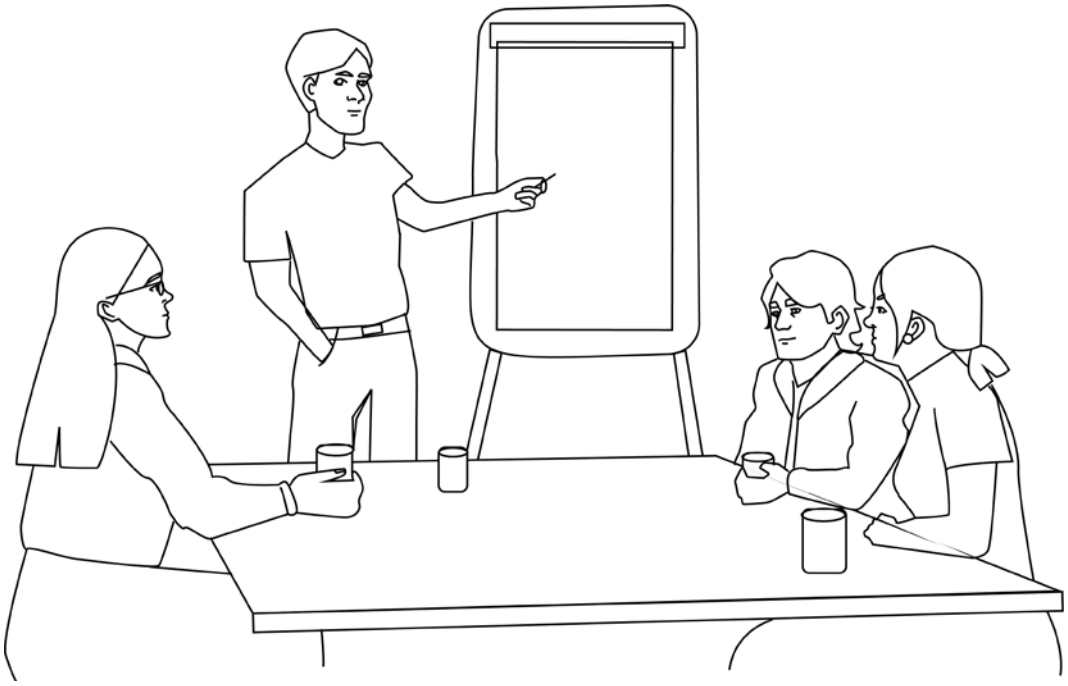
- **Pozytywna współzależność**: cały zespół ustala zasady, zgodnie z którymi członkowie będą pracowali dla osiągnięcia wspólnego celu; wszyscy członkowie czują się odpowiedzialni za siebie i za zespół.
- **Odpowiedzialność indywidualna**: działania każdego członka mają bezpośredni wpływ na cały zespół. Wpływ ten może być pozytywny lub negatywny. Każdy członek zespołu czuje się odpowiedzialny za realizację swojego zadania na rzecz wspólnego celu.
- **Równe uczestnictwo**: wszyscy członkowie mają równe szanse na uczestnictwo w pracach zespołu. Bardzo ważną częścią tej zasady jest to, że praca powinna być sprawiedliwie dzielona od samego początku. Jedna osoba nie powinna mieć znacząco więcej pracy niż inna.
- **Jednoczesna interakcja**: wszyscy członkowie zespołu muszą się ze sobą komunikować, to znaczy: dzielić się swoimi opiniami, wyrażać emocje dotyczące pracy i wspólnie podejmować decyzje. Jeśli brakuje interakcji, zespół może się rozpaść, co uniemożliwi osiągnięcie celu.

Zwieńczeniem pracy zespołu jest **refleksja**, czyli **przetwarzanie grupowe**, gdy po zakończeniu wspólnej pracy nad zadaniem projektowym członkowie zespołu dzielą się ze sobą wrażeniami o tym, jak im się współpracowało, co było dobre, a co nie. To bardzo ważny moment całego procesu, pozwalający przyswoić, utrwalić i zintegrować nowo nabyte umiejętności społeczne.

Podsumowując: uczenie się we współpracy w porównaniu z metodą tradycyjną daje uczniom możliwość bardziej efektywnego przyswajania wiedzy z konkretnej dziedziny jak i nabywania umiejętności z obszaru rozwoju osobistego.

4 Cooperative learning, S.Kagan, M. Kagan, Kagan Publishing 2015

Według badań przywołanych we wspomnianej wyżej pracy uczniowie uczący się we współpracy:



- mają większe zdolności rozwiązywania problemów i wykazują rozwinięte strategie kognitywnego[5] rozumowania,
- lepiej zapamiętują i przekazują skomplikowany materiał,
- lepiej posługują się technologią komputerową,
- lepiej rozumieją tekst, pisownię i werbalną ekspresję,
- zyskują większą samodzielność i jednocześnie rozwijają zachowania kooperatywne,
- wykazują większą tolerancję, więcej zachowań wspomagających i koleżeńskich wobec przedstawicieli innych ras/grup etnicznych lub odmiennej płci,
- mają wyższy poziom zrozumienia i tolerancji wobec osób z niepełnosprawnościami i wykluczonych społecznie,
- mają wyższe poczucie własnej wartości,

5 kognitywny «mający związek z poznawaniem świata lub badaniem procesów poznawczych», Słownik języka polskiego PWN.

- mają niższy poziom lęku i stresu,
- wykazują większy poziom zadowolenia.

Uczenie się we współpracy bardzo dobrze koresponduje z zasadami PBL *(Problem Based Learning)*, czyli uczenia się poprzez rozwiązywanie problemów. Według twórcy PBL Howarda Barrowsa[6] metodę charakteryzuje sześć elementów:

- koncentracja na odbiorcy – uczniu,
- uczniowie pracują w małych zespołach,
- rozwiązywane problemy/zadania mają wymiar praktyczny i odniesienie do życia codziennego,
- uczniowie współpracując ściśle z innymi członkami zespołu rozwijają kompetencje społeczne,
- uczniowie zdobywają wiedzę, narzędzia i informacje zarówno poprzez pracę zespołową jak i indywidualną.
- nauczyciel wspomaga proces komunikacji wewnątrz zespołów – jest facylitatorem.

Jest to metoda skoncentrowana na odbiorcy – czyli uczniu. Proces nauczania według filozofii PBL jest ściśle związany z obecnością problemu, zadania, które należy rozwiązać. Wiedza jest ukryta w zadaniu, a cele kształcenia są realizowane podczas prac nad jego zrealizowaniem. Uczniowie w zespołach pracują nad rozwiązaniem praktycznego zagadnienia, znajdującego swoje miejsce w realiach dnia codziennego. Zapotrzebowanie na taką właśnie formułę wybrzmiało dobitnie w wypowiedziach uczniów podczas wywiadów, przeprowadzonych w trakcie przygotowywania niniejszego przewodnika.

Zainteresowanych metodą *Problem Based Learning* zachęcam do lektury artykułu[7] omawiającego metodę, rezultaty jej stosowania oraz podającego piśmiennictwo.

6 Problem-Based Learning: An Approach to Medical Education, Howard Barrows, 1980
7 Problem-Based Learning, Jakub Szczepaniak, Iwona Wróblewska, Forum Akademickie nr 07-08/2012, https://prenumeruj.forumakademickie.pl/fa/2012/07-08/problem-based-learning/, dostęp 09.01.2022

Droga Nauczycielko, Drogi Nauczycielu, aby wszystkie te elementy faktycznie zadziałały i dały trwałe efekty, Twoja rola staje się nieco odmienna od tej w tradycyjnym modelu nauczania. Zainteresowanych pogłębieniem tematu zachęcam do lektury artykułu i książki Spencera Kagana.

Poniżej, w skrócie, cztery oblicza Twojej roli.

Planista – uczenie się we współpracy wymaga od Ciebie, abyś miał wizję i plan tego, co będzie się działo na lekcjach, a także zakresu samodzielnej pracy ucznia poza czasem lekcyjnym. Musisz także zaplanować i zabezpieczyć całą techniczną stronę zajęć.

Tutor/facylitator – podczas wstępnych zajęć przeprowadzasz proces podziału na zespoły, omawiasz zasady pracy i oceniania, określasz czas, w którym należy zakończyć pracę itp. Przedstawiasz także siebie w roli tutora/facylitatora – to ważne dla klarowności sytuacji. Zachęcasz też uczniów do stworzenia kontraktów w zespołach, takich, które ułatwią im wspólne działanie. Zadania i role w zespole uczniowie przydzielają sobie sami, bo to ważna część procesu, ale może się zdarzyć, że Twoje wsparcie i rada będzie niezbędna. Warto też upewnić się, że zadania do wykonania rozdzielone zostały równomiernie pośród członków zespołu.

Podczas lekcji obserwujesz interakcje uczniów – dzięki temu wiesz, które zespoły potrzebują Twojego wsparcia.

Arbiter – o sytuacjach konfliktowych jeszcze będziemy mówić. One będą się zdarzać, jak zwykle w życiu. Tu jedynie wspomnę, że to dla Ciebie doskonała okazja do pokazania skuteczności otwartej komunikacji i szacunku dla stron, empatii i przy okazji, do ugruntowania swojego autorytetu.

Ewaluator – po zakończeniu pracy będziesz ją oceniał w dwóch wymiarach:

- technicznym (osiągnięcie założonego celu, poprawność funkcjonalna rozwiązania, pozytywny code review etc.)
- społecznym (samoorganizacja zespołu, wykonywanie zadań w ramach przypisanych ról, branie odpowiedzialności za siebie i zespół, umiejętności komunikacji i rozwiązywania konfliktów etc.).

Przed rozpoczęciem pracy zespołów i po jej zakończeniu użyjesz przygotowanego przez nas testu kompetencji społecznych i na tej podstawie będziesz mógł ocenić efektywność całego procesu.

Warto tu przypomnieć, że nie ma lepszego sposobu wsparcia dla utrwalenia wiedzy i umiejętności, niż dobre słowo i pochwała w odpowiednim momencie. Zakończenie pracy to także wyśmienita okazja do stworzenia przestrzeni dla zespołów, aby mogły skomentować projekty koleżanek i kolegów, a także dla wszystkich uczniów, aby mogli ocenić zajęcia, sposób ich prowadzenia, co im pomagało, co utrudniało pracę, itd. To także doskonała okazja dla Ciebie, żeby dostać informację zwrotną od swoich uczniów.

Zespoły

Skład zespołów ustalisz Ty, w oparciu o wyniki krótkiego testu oraz własnej znajomości uczniów. Uczniowie nie mają wpływu na ten proces. Myślą przewodnią jest, aby uczniowie nauczyli się pracować w zespołach, na których dobór nie mają wpływu.Taka sytuacja będzie im często towarzyszyć w pracy, którą podejmą jako dorośli. Jest to więc dla nich doskonały poligon, możliwość przećwiczenia w bezpiecznych warunkach szkolnych, umiejętności współpracy z osobami, które niekoniecznie darzą największą sympatią.

S. Kagan w swojej książce i w licznych artykułach przedstawia optymalny, według niego, skład zespołu.

Zespoły powinny być heterogeniczne (mieszane) pod względem poziomu umiejętności, płci i typów osobowości, czyli jak najbardziej różnorodne. Za takim podejściem przemawia kilka argumentów:

- jeśli w każdym zespole jest osoba o wysokich umiejętnościach to najprawdopodobniej będzie ona w stanie wyjaśnić pozostałym ewentualne niejasności we wskazówkach nauczyciela lub w pomocniczym materiale wideo i utrzymać zespół w zadaniu. Znakomicie poprawia to płynność zarządzania klasą podczas zajęć;
- w przypadku zespołów jednorodnych pod względem poziomu umiejętności istnieje niebezpieczeństwo powstania zespołów zwycięzców i przegranych, i tym samym powstania problemów klasowych związanych ze statusem i wzajemnym szacunkiem;

- im więcej różnorodności, tym więcej uczniowie mogą się od siebie nauczyć;
- tzw. korepetycje rówieśnicze to często najskuteczniejszy sposób nauki dla mniej zdolnych członków zespołu. Dodatkowym zyskiem jest to, że "dający korepetycje" często zyskuje przynajmniej tyle samo, co biorący je;
- zespoły heterogeniczne poprawiają umiejętności społeczne u wszystkich, zwłaszcza u osób osiągających dobre wyniki. Zwykle nie mają one problemu z bardziej skomplikowanymi treściami. Ich największym obszarem rozwoju są często relacje międzyludzkie - gdy uczniowie pracują w mieszanych zespołach, uczą się przekazywania wiedzy, cierpliwości, zachęcania;
- optymalna liczba osób w zespole według Kagana to cztery, bo czteroosobowe zespoły maksymalizują i wyrównują aktywne uczestnictwo w porównaniu z każdą inną liczebnością. Zainteresowanych odsyłam do artykułu[8]. Również Dr Meredith Belbin, autor teorii ról w zespole, uważa, że idealna wielkość zespołu to cztery osoby[9] - oczywiście, jeśli to możliwe, w parytecie płci, czyli dwie dziewczyny i dwóch chłopców. Jeśli nie ma takiej możliwości, to zespół powinien być jednorodny pod względem płci. I jeszcze raz oczywiste jest, że czasem nie ma innej możliwości i zespół będzie liczył np. 3 osoby.

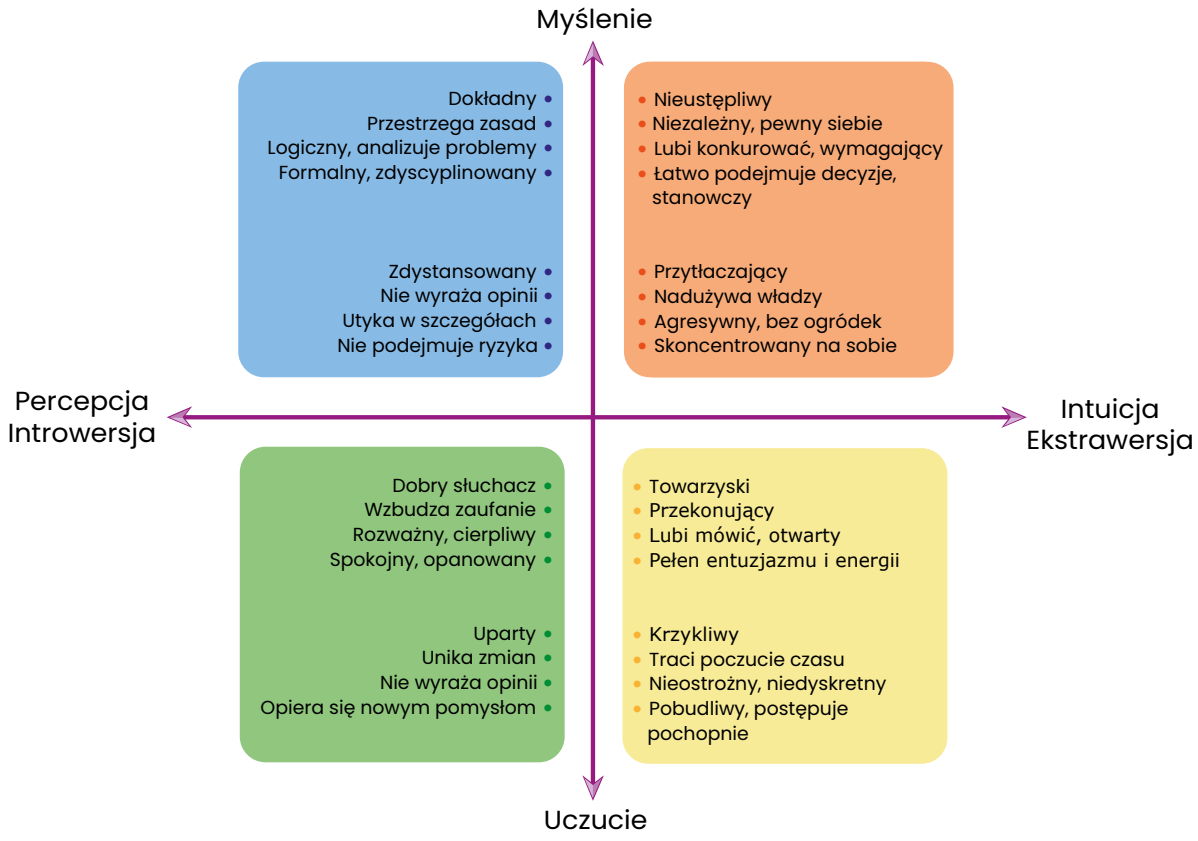
No to dzielimy ...

Pierwszym elementem w tym procesie będzie wynik krótkiego testu predyspozycji. Test został opracowany przez Wiktora Wołoszkę i z powodzeniem wykorzystany w jednej z jego gier szkoleniowych[10].

Test bazuje na modelu stworzonym przez Carla Gustava Junga, jednego z twórców podwalin dla współczesnej psychologii, rozwiniętym i udoskonalonym przez wielu badaczy, zgodnie z którym ludzie posiadają odmienne predyspozycje, przez co postrzegają te same sytuacje z różnych perspektyw.

8 https://www.kaganonline.com/free_articles/dr_spencer_kagan/295/Teams-of-Four-Are--Magic!
9 https://www.belbin.pl/role-zespolowe-belbina/, dostęp 11.01.2022
10 GraSzkoleniowa.pl, Wiktor Wołoszko

Przyjęcie do wiadomości, zrozumienie tej odmienności, jest kluczowe dla efektywnego komunikowania się i współpracy w zespole. Żeby efektywnie porozumiewać się i współpracować z innymi ludźmi, trzeba najpierw spróbować spojrzeć na świat z ich perspektywy.

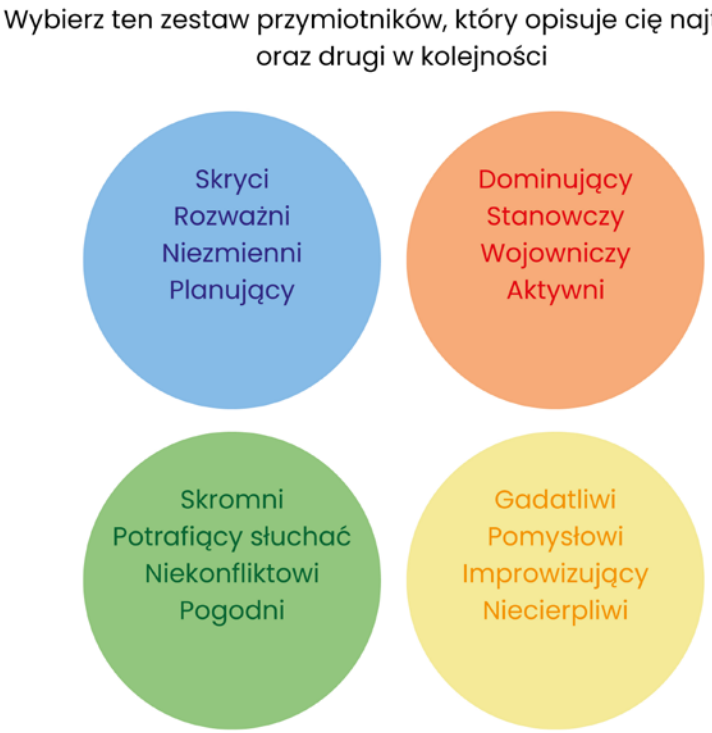


Współcześnie obowiązujący model obejmuje cztery obszary, przedstawiające cztery podstawowe predyspozycje ludzkich zachowań. Obszary te to ćwiartki układu utworzonego z dwóch przecinających się prostopadłych osi: Percepcja – Intuicja oraz Myślenie – Uczucie. Przykładową graficzną ilustrację modelu przedstawia rysunek. Obszary przypisane poszczególnym predyspozycjom oznaczono kolorami: czerwonym, żółtym, zielonym i niebieskim. Literatura opisująca szczegółowo model jest szeroko dostępna. Obrazowo i przystępnie, w kontekście efektywnej komunikacji przedstawiono go tutaj[11].

W naszym teście powyższymi kolorami oznaczono grupy przymiotnikówopisujących zachowania korespondujące z predyspozycjami z modelu Junga.

11 https://4grow.pl/taxonomy/term/5170

Każdy z uczniów wybiera ten zestaw przymiotników, który według niego samego najtrafniej go opisuje oraz drugi w kolejności. W idealnej sytuacji każdy członek zespołu reprezentuje inny typ predyspozycji. W przypadku nierównomiernego rozkładu typów w klasie bierzemy pod uwagę drugi wbór.



Przypominam, że podziału na zespoły dokonujesz Ty, Droga Nauczycielko, Drogi Nauczycielu, na podstawie powyższego testu oraz Twojej znajomości uczniów, ich poziomu umiejętności i wiedzy, z uwzględnieniem parytetu płci. Zadbaj więc, aby w jednym zespole, tak jak zaleca Kagan, nie znalazły się same osoby najbardziej lub najmniej zaawansowane, jeśli chodzi o wiedzę i umiejętności dotyczące programowania. Będziesz więc potrzebował na to chwili czasu do zastanowienia. Na pewno pomoże w tym przygotowana (np. w Excelu) lista uczniów zawierająca wynik testu (1 i 2 wybór), twoją ocenę informatycznego zaawansowania, płeć i ewentualnie inne czynniki, które uznasz za istotne (np. ryzykowne jest aby w jednym zespole znalazły się osoby silnie dominujące lub jawnie ze sobą skonfliktowane).

Wyłoniony przez Ciebie zespół będzie potrzebował zapewne trochę czasu, aby się zintegrować. Zachęć uczniów do kontaktu, spotkania w czasie pozalekcyjnym. Zaproponuj, aby zespoły nadały sobie nazwy, np. w nawiązaniu

do tematu swojej pracy – to ważny moment integrujący, dający poczucie identyfikacji i zaspokajający potrzebę przynależności.

Zasady pracy - kontrakt

Sformułowanie i przekazanie uczniom zasad, według których będzie przebiegała praca w ramach projektu, to ważna część procesu budowania ich świadomej odpowiedzialności, poczucia podmiotowości oraz nowego wymiaru relacji między Tobą a nimi, opartej na wzajemnym szacunku. Część tych zasad ma charakter bardziej formalny, wynikający z przyjętej formuły zajęć i treść tej części kontraktu zaproponujesz Ty (znajdziesz je w dalszej części tekstu), natomiast bardzo ważne jest, aby wszystkie pozostałe zasady zostały ustanowione wspólnie a uczniowie mogli wnieść swoje zapisy. Wtedy każda ze stron może się do takiego kontraktu odwołać dzięki elementarnej identyfikacji z jego treścią, jeśli zaistnieje wymagająca tego sytuacja. Podczas dyskusji postaraj się spojrzeć na problemy również z perspektywy ucznia.

Poniżej kilka sugestii dotyczących samego procesu:

- pokaż sens kontraktu - spisujemy go po to, aby nasza praca podczas zajęć była efektywna, przebiegała możliwie gładko a cel każdego zespołu mógł być osiągnięty.
- pokaż, że obie strony umawiające się mają wpływ na zasady pracy. Zapytaj po prostu co, ich zdaniem, powinno zostać zapisane. Przedyskutujcie to i podejmijcie stosowne decyzje.
- zaproponuj kategorię zasad, które znajdą się w kontrakcie, np.:
- co robimy - szanujemy się, angażujemy, wykonujemy swoje zadania, itp
- czego nie robimy - nie spóźniamy się, nie przeszkadzamy innym zespołom, itp.

Możesz też skorzystać z sugestii zawartych w książce Ewy Góralczyk[12].

Kontrakt warto podpisać, bo swoim podpisem potwierdzam: tak, zgadzam się na to, co jest zapisane. Podpisany kontrakt dobrze jest umieścić w sali, w której odbywają się zajęcia, po to, żeby zawsze można było do niego się-

12 E. Góralczyk, Umowa z klasą. Wyd. Fraszka Edukacyjna, Warszawa 2015.

gnąć.

Poniżej formalne zasady obowiązujące podczas zajęć „Progr@muj w zespole”:

- Uczennice/uczniowie pracują w zespołach. Skład zespołów ustala nauczyciel na podstawie testu i swojej znajomości uczniów.
- Każdy zespół samodzielnie wybiera temat swojego zadania i jest odpowiedzialny za jego wykonanie.
- Przed przystąpieniem do pracy uczennice/uczniowie wypełniają test umiejętności współpracy i komunikacji w zespole.
- Uczennice/uczniowie otrzymują materiały wspierające: przewodnik dla uczniów, materiały wideo.
- Czas trwania zajęć to 7 godzin lekcyjnych podczas zajęć szkolnych i co najmniej 5 godzin pracy własnej w czasie poza lekcyjnym.
- Uczennice/uczniowie w każdym zespole wspólnie dokonują podziału pracy koniecznej do wykonania zadania. Każdy członek zespołu musi być zaangażowany i ma do wykonania swoją część zadania.
- Zespoły pracują samodzielnie.
- Nauczyciel służy radą i wsparciem jeśli zostanie o niepoproszony.
- Uczennice/uczniowie mają prawo do swoich opinii dotyczących zadania i ponoszą pełną odpowiedzialność za skutki podjętych przez zespół decyzji.
- Po zakończeniu pracy nad zadaniem uczennice/uczniowie wykonują ponownie test umiejętności współpracy i komunikacji w zespole.
- Wykonana praca podlega ocenie przez nauczyciela. Ocenie podlega osiągnięty rezultat programistyczny oraz praca uczniów w zespole.

Kontrakt może Cię również zabezpieczyć przed ewentualnym przeniesieniem zasad pracy z projektu na pozostałe lekcje prowadzone w sposób tradycyjny, jeżeli taka będzie twoja intencja. Możesz wtedy wpisać klauzulę, że zapisy dotyczą tylko i wyłącznie pracy podczas zajęć „Progr@muj w zespole”.

W tym miejscu raz jeszcze przypominamy, abyś zachęcił uczniów do zawarcia osobnych kontraktów w zespołach. To istotny element w procesie przyjmowania odpowiedzialności zarówno za siebie jak i za zespół, a jednocześnie ugruntowanie poczucia przynależności i identyfikacji z zespołem. Te kontrakty uczniowie powinni sformułować samodzielnie, aczkolwiek Twoja rada, jeśli o nią poproszą, może być cenna. Przykład takiego kontraktu znajdziesz poniżej.

KONTRAKT

- ROBIMY WSZYSTKO, ABY NASZ CEL ZOSTAŁ OSIĄGNIĘTY,
- DOTRZYMUJEMY SŁOWA, WYKONUJEMY NA CZAS SWOJE ZADANIA,
- REALIZUJEMY PODJĘTE DECYZJE,
- PRZYCHODZIMY PUNKTUALNIE NA SPOTKANIA,
- KOMUNIKUJEMY SIĘ WEDŁUG USTALONYCH ZASAD,
- SYGNALIZUJEMY ZESPOŁOWI PROBLEMY Z WYKONANIEM SWOJEGO ZADANIA, W RAZIE POTRZEBY PROSIMY O POMOC,
- W RAZIE POTRZEBY POMAGAMY SOBIE ALE NIE WYRĘCZAMY W PRACY,
- W PRZYPADKU ODMIENNYCH OPINII ZNAJDUJEMY KONSENSUS,
- KAŻDEGO SŁUCHAMY Z TAKĄ SAMĄ UWAGĄ, NIE PRZERYWAMY MÓWIĄCEMU,
- SZANUJEMY SIĘ, NIE OBRAŻAMY NIKOGO, NIE ROBIMY FOCHÓW,
- KONFLIKTY STARAMY SIĘ ROZWIĄZYWAĆ SAMI, W RAZIE POTRZEBY PROSIMY NAUCZYCIELA O POMOC.

Spisany kontrakt powinien zostać podpisany przez wszystkich członków zespołu.

Role w zespole

Zwykle, gdy pojawia się temat podziału i objęcia ról w zespole, pojawia się również nazwisko wspomnianego już Mereditha Belbina. Opracował on teorię ról w zespole i dowiódł, że nie tylko wiedza i umiejętności, ale również typ osobowości, preferencje i styl komunikacji mają wpływ na efektywność zespołów[13]. Nie jest naszą intencją przytaczanie tutaj teorii Belbina i jedynie wspomnimy, że na podstawie badań i obserwacji pracy różnych zespołów opisał on trzy profile zachowań ich członków. Są to:

- role zadaniowe – osoby, które w pracy w zespole mają silne ukierunkowanie na realizację powierzonych zadań,
- role socjalne, zorientowane na ludzi – osoby, które najlepiej funkcjonują i działają, gdy mają obok innych członków zespołu,
- role intelektualne – osoby, których zachowanie jest ukierunkowanie na myślenie, rozważanie oraz analizowanie.

W ramach tych trzech profili Belbin wyróżnił i nazwał 9 ról zespołowych. Są one według Belbina stałe i oznaczają skłonność do określonych zachowań, nawiązywania współpracy i kontaktów oraz komunikowania się we właściwy danej osobie sposób. Jedna osoba zwykle może przyjąć dwie, trzy role.

Predyspozycje do przyjmowania tych ról można zbadać przy pomocy odpowiedniego, dość rozbudowanego testu. W sytuacji, gdy mamy potrzebę, możliwość i swobodę kompletowania zespołu, informacja o predyspozycjach potencjalnych kandydatów pozwala na taki ich wybór, aby wszystkie role zostały obsadzone.

W naszym pomyśle na zajęcia, w sposób zamierzony odtwarzamy sytuację, jaką często można spotkać w realiach mniejszych organizacji, gdy zespół ma do wykonania określone zadanie, a możliwość swobodnego doboru jego członków jest ograniczona, przypadkowa lub wcale jej nie ma.

13 M. Belbin, Twoja rola w zespole (Team Roles at Work), Gdańskie Wydawnictwa Psychologiczne, Gdańsk 2008

Wtedy role tylko częściowo mogą być zgodne z preferencjami, a pewna ich część jest wymuszona sytuacją.

Dlatego tak ważna jest różnorodność predyspozycji wśród członków, o której wspomnieliśmy przy okazji podziału na zespoły. Zwiększa ona po prostu szanse na objęcie większości zidentyfikowanych ról i w rezultacie, na efektywną pracę zespołu.

Role zostaną objęte przez uczniów samoistnie podczas pierwszego spotkania zespołu. Być może zostaniesz poproszony o radę, ale zachęcamy do powściągliwości, aby nie zwalniać zainteresowanych z odpowiedzialności za podejmowane decyzje.

Uczniowie muszą samodzielnie rozdzielić między siebie zadania do wykonania. Część z tych zadań będzie bardziej zgodna z ich naturalnymi preferencjami i predyspozycjami, a część mniej, tak jak to zwykle bywa w życiu.

Twoją rolą, Drogi Nauczycielu, Droga Nauczycielko, jest monitorowanie, aby każdy członek zespołu miał przydzielone zadania i był za nie odpowiedzialny.

Konflikt

Konflikty mogą pojawić się w każdym momencie i w każdej zbiorowości ludzkiej. Są po prostu naturalnym procesem społecznym zachodzącym między ludźmi. Najczęściej mają swoje źródła w różnych wyznawanych wartościach, sprzecznych poglądach, celach lub interesach.

Konflikty w zespole mogą mieć charakter konstruktywny lub destrukcyjny.

Charakter konstruktywny mają np. spory o możliwe rozwiązania problemu czy zadania, prezentacja sprzecznych opinii i poglądów. I jeśli są one traktowane jako zbiór dostępnych rozwiązań, to prowadzą w efekcie do stworzenia nowych jakości i mogą być pozytywnym bodźcem w procesie osiągania celu zespołowego.

Konstruktywne rozwiązywanie konfliktów prowadzi do optymalnych rozwiązań typu wygrany-wygrany. W tym procesie jest przestrzeń na zastosowanie wszelkich dostępnych metod: negocjacji, mediacji, facylitacji i arbitrażu.

Osiągnięcie porozumienia zawsze jest najlepszym rozwiązaniem. Czasami jest ono poprzedzone długimi negocjacjami pochłaniającymi czas i energię zespołu ale za to przynosi znaczące korzyści w przyszłości. Jeżeli wszyscy członkowie zespołu wypracowywali decyzję, to jest ona przez wszystkich akceptowana i może zostać sprawnie wprowadzona w życie. Warto ten mechanizm pokazać uczniom i zachęcić, aby w przypadku zaistnienia spornych kwestii wszyscy członkowie zespołu angażowali się w osiągnięcie konsensusu.

Poniżej kilka wskazówek, które im w tym pomogą:

- Słuchaj uważnie i dopytuj o przyczyny odmiennego stanowiska, co naprawdę za nim stoi. Bądź otwarty na wszystkie pojawiające się idee.
- Wyjaśnij na początku dyskusji dlaczego, jako zespół, musicie teraz podjąć decyzję i czego ona będzie dotyczyć.
- Zachęcaj wszystkich członków zespołu do aktywnego udziału w dyskusji. Nie zakładaj, że milczenie oznacza zgodę.
- Postrzegaj różnorodność opinii i stanowisk jako rzecz naturalną i pozytywną, zwiększającą pulę danych, którą zespół może użyć w procesie podejmowania decyzji.
- Nie ulegaj pokusie porzucenia swojego sposobu myślenia tylko po to, żeby uniknąć konfliktu.
- Nie ulegaj pokusie wyszukiwania jedynie takich argumentów, które potwierdzą twój punkt widzenia. Szukaj takich, które będą łącznikiem do pomysłów innych członków zespołu.
- Jeśli masz większą wiedzę w temacie niż pozostali członkowie zespołu, zachowaj powściągliwość i poczekaj do momentu aż oni się wypowiedzą.
- Zadbaj abyście mieli wystarczająco dużo czasu na dyskusję i podjęcie przemyślanej decyzji.
- Sprawdź i poproś o to pozostałych, czy każdy rozumie na czym polega wasza decyzja, dlaczego jest najlepsza.

Dużo trudniejsze w opanowania są konflikty destrukcyjne, które zwykle są spowodowane wrogim działaniem, złością, poczuciem skrzywdzenia, nie-

uszanowania konkretnego członka zespołu. Czasami też bywają one przeniesione z obszaru towarzyskiego. O tego rodzaju konflikcie warto chwilę porozmawiać przed rozpoczęciem pracy i uświadomić uczniom, że się może pojawić oraz, że zlekceważenie go lub ukrycie redukuje szansę na efektywną pracę zespołu. Sposób postępowania w przypadku jego zaistnienia można zapisać w kontrakcie i warto to zasugerować uczniom.

A poniżej garść wskazówek, prosto ze szkoły[14], które być może, Drogi Nauczycielu, Droga Nauczycielko, przydadzą Ci się w przypadku mediowania między stronami takiego destrukcyjnego konfliktu:

- słuchaj uważnie dwóch stron, bezstronnie, nie zajmuj stanowiska,
- słuchaj aktywnie, utrzymuj kontakt wzrokowy ze stronami konfliktu w zależności, która strona mówi, parafrazuj, dopytuj, odzwierciedlaj uczucia,
- dowartościowuj, np. to, co mówisz, jest ważne...,
- porządkuj, np. do tej pory ustaliliśmy, że…,
- zadaj pytanie: jak chcielibyście zakończyć konflikt?,
- pytaj zawsze, jakie są oczekiwania wobec strony przeciwnej,
- zachowuj neutralność, nie narzucaj rozwiązania, akceptuj przyjęte rozwiązania i pomagaj zastanowić się, czy przyjęte rozwiązania są wykonalne,
- pytaj obie strony konfliktu, co sądzą o sposobie jego zakończenia,
- jeżeli strony chcą się przeprosić, spytaj jak rozumieją słowo przepraszam, bo ważne, żeby przepraszanie było autentyczne,
- obserwuj strony konfliktu, jeżeli widzisz, że między uczniami jest napięcie, to dopytaj bez zniecierpliwienia, co jeszcze jest do wyjaśnienia?,
- jeżeli nie masz czasu spytaj: czy możemy z tym poczekać np. do jutra?,
- jeżeli masz czas i emocje opadły, spytaj uczniów, jak inaczej mogli się zachować, żeby do konfliktu nie doszło.

14 http://sp27.kielce.eu/content/rozwiazywanie-konfliktow, dostęp 12.01.2022

Słów kilka o komunikowaniu się

Podstawową, niezbędną umiejętnością, umożliwiającą rozwijanie innych kompetencji społecznych jest umiejętność komunikowania się czyli tzw. efektywna komunikacja interpersonalna.

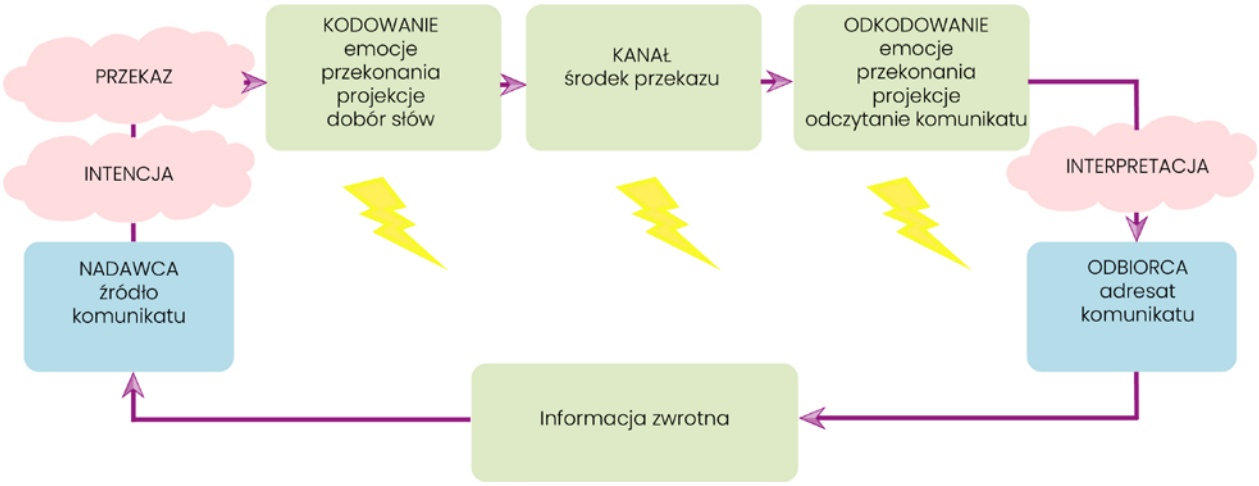
Komunikowanie się to psychologiczny proces, w którym przekazujemy i otrzymujemy informacje kontaktując się z drugą osobą. Tak naprawdę wszystko w tym kontakcie jest komunikacją. Nasze myśli, opinie czy relacje przekazujemy słowami, tonem głosu, mimiką, postawą ciała, ubiorem i nie tylko. Samo mówienie nie wystarczy, aby się skutecznie komunikować. Komunikując się z drugim człowiekiem możesz mówić, słuchać i pytać. Zachęcamy, abyś na własny użytek przeprowadził autodiagnozę, jaki procent w Twojej codziennej komunikacji ma mówienie, słuchanie i zadawanie pytań. To oczywiste, że będąc nauczycielem dużo mówisz. A jak dużo słuchasz (z ciekawością i otwartością na rozmówcę)? Jak wiele pytań zadajesz rozmówcy po to, żeby go lepiej poznać i zrozumieć to, co on chce przekazać, żeby dostrzec jego punkt widzenia?

Myślę, że taka autorefleksja jest ważna wobec faktu, że umiejętność słuchania i zadawania pytań jest tą, którą chcemy rozwinąć wśród uczniów podczas trwania naszych zajęć. Zwiększenie udziału uważnego słuchania i zadawania pytań może przynieść zaskakujący wzrost efektywności komunikacji.

Komunikacja z drugim człowiekiem zaczyna się od intencji. Intencja jest tym, co nadawca komunikatu chce osiągnąć bardziej lub mniej świadomie. Intencja to zamysł, myśl przewodnia, pragnienie. Niewłaściwe odczytanie intencji z jednej strony i niespełnione oczekiwania z drugiej gwarantuje nam komunikacyjny klops.

Bycie świadomym swojej intencji jest niezbędne do tego, aby nasze myśli były ukierunkowane na efektywne przekazanie sensu a słowa wyraziły dokładnie to, o co nam konkretnie chodzi. Niestety, często w tym momencie pojawiają się emocje, nasze myśli wpadają w błędne koło przekonań, projekcji, nadinterpretacji i obaw, że odbiorca nas nie zrozumie, źle o nas pomyśli i w efekcie zniekształcają nasz przekaz. Ten proces nazywamy kodowaniem naszego komunikatu. Po drugiej stronie jest odbiorca ze swoim analogicznym bagażem projekcji, przekonań i nadinterpretacji, emocji, który

usiłuje odkodować nasz przekaz. W efekcie komunikat nadany i odebrany to często zupełnie różne treści.



Kodowanie jest najbardziej krytycznym momentem komunikacji. Albo nadawca utrzyma kontakt ze swoją intencją, będzie jej świadomy i powie, o co konkretnie mu chodzi, albo znacząco obniża szansę na porozumienie. Żeby tego kontaktu z intencją nie tracić, możesz świadomie używać w swojej komunikacji takich sformułowań jak:

Moją intencją jest, aby …

Chcę zadbać o to, aby …

Wartością dla mnie w tym jest to, że …

Zależy mi na tym,aby …

Taka sama odpowiedzialność za efektywność komunikacji jest po stronie odbiorcy – uważne słuchanie i zadawanie pytań znacząco zwiększa szanse na porozumienie. I tu możesz świadomie używać takich sformułowań jak:

Co jest twoją intencją?

O co chcesz zadbać?

Co w tym dla ciebie jest wartościowe?

Na czym ci zależy?

Pierwszym krokiem do skutecznej komunikacji jest chęć porozumienia się po obu stronach. Nie warto zakładać, że rozmówca zrozumiał, ale że chce

zrozumieć. Warto pamiętać, że jako nadawca i odbiorca jednocześnie, każdy z nas jest współodpowiedzialny za efekt komunikowania się.

Dlatego tak ważne jest pogłębianie samoświadomości i kontakt ze swoimi emocjami.

Gdy pojawiają się emocje...

…to znaczy, że dzieją się rzeczy ważne. Jeśli to Twoje emocje, to są to rzeczy ważne dla Ciebie. Jeśli to emocje innej osoby, ucznia albo całej grupy uczniów, klasy, to znaczy, że to są rzeczy ważne dla niej/nich. Tylko od Ciebie zależy, jak je odczytasz, jak nimi zarządzisz, czy skorzystasz z informacji, które ci przynoszą. Emocje mają to do siebie, że pojawiają się dosłownie w jednej chwili. Radzenie sobie z emocjami nie polega na ich tłumieniu i kontrolowaniu, tylko na ich doświadczaniu i przeżywaniu. Dlatego tak ważna jest umiejętność wyrażenia, jak się w danym momencie czujesz, nazwania, opisania swoich emocji. Okazuje się, że najlepszym sposobem na wyciszenie emocji w trudnej sytuacji jest uznanie ich realności, nazwanie ich:

tak, jestem niezadowolona,

tak jestem rozczarowany,

tak, widzę, że się denerwujesz.

Zaprzeczanie im:

nie ma sensu tak się denerwować, nie powinnaś czuć się dotknięta,

nie bądź taki urażony,

nie ma co robić takiej tragedii,

powoduje tylko aktywację mechanizmów obronnych, prowadzi do frustracji, złości, jest odbierane jako lekceważenie, brak szacunku i osądzanie.

Potwierdzenie, nazwanie emocji ma sens ponieważ:

- czyjeś uczucie, emocja jest po prostu faktem,
- sygnalizuje coś ważnego kryjącego się pod tą emocją,
- dotarcie do tego czegoś, do rzeczywistego problemu, możliwe jest dopiero wtedy, gdy emocja zostanie uznana i zauważona,

- poszanowanie czyichś emocji oznacza okazanie szacunku osobie te emocje przeżywającej, a to jest podstawą dobrej relacji.

Emocje działają jak kierunkowskaz: zmierzamy w kierunku tych przyjemnych, wzmagających chęć życia i działania, natomiast aktywnie unikamy tych nieprzyjemnych, osłabiających, destrukcyjnych. Emocje mają więc wartość motywacyjną; można by powiedzieć, że motywacja to emocja[15].

Emocje są też informacją, czy podążamy za tym, co jest dla nas ważne, za swoimi wartościami, czy działamy wbrew sobie? Czy nasze potrzeby są zaspokojone, czy też nie? Wiele potrzeb jest wspólnych dla wszystkich ludzi. Należą do nich potrzeby: akceptacji, docenienia, wysłuchania i bycia zauważonym.

Myślę, że zgodzisz się, że są to potrzeby uczniów, dla których Ty jesteś tutorem. Aby te potrzeby mogły zostać zauważone i zaspokojone, w Waszej relacji musi zostać zbudowane zaufanie. Zaufanie buduje się w oparciu o kluczowe wartości: szacunek, życzliwość, odpowiedzialność.

Zaufanie jest niezbędne dla efektywnego komunikowania się.

Dlatego Twoja rola, Droga Nauczycielko, Drogi Nauczycielu jest po prostu kluczowa.

15 R. Gut, A. Gut, Pokolenie Y. Zarządzanie sobą na Zielonej Ścieżce. Wyd. Instytut Flashpoint, Wrocław 2016,

Badanie umiejętności współpracy i komunikacji w zespole

BADANIE UMIEJĘTNOŚCI WSPÓŁPRACY I KOMUNIKACJI W ZESPOLE

WPROWADZENIE

Poniższy test został przygotowany na bazie opracowania Science Museum of Minnesota (SMM)[1]. Zamysł testu koncentruje się na fakcie, że kluczową kompetencją, na której opiera się współpraca w zespole, jest skuteczne komunikowanie się.

Wyróżniliśmy pięć obszarów w komunikacji w zespole:

1. dzielenie się pomysłami i informacjami (pytania 1-6)
2. precyzja przekazu (pytania 7-12)
3. asertywność (pytania 13-18)
4. otwartość i uważność w zadawaniu pytań (pytania 19-24)
5. koncentracja i uważne słuchanie (pytania 25-28)

O znaczeniu zespołu i wymienionych tutaj umiejętności możesz przeczytać więcej w *Przewodniku dla nauczycieli po kompetencjach współpracy i komunikacji w zespole* oraz źródłach zewnętrznych. Pamiętaj: ucząc przedmiotu, w tym wypadku informatyki, stosuj elementy pracy zespołowej - kompetencje społeczne są często kluczowym aktywem absolwentów przy poszukiwaniu zatrudnienia i są bardzo pożądane przez pracodawców.

Struktura testu

Test ma formułę samooceny badanego i mierzy łatwość, komfort oraz prawdopodobieństwo użycia konkretnej umiejętności w określonej sytuacji. Te sytuacje są wkomponowane w scenariusz pracy zespołowej, będący kontekstem dla zadanych pytań.

Wszystkie 28 pytań ma opcje odpowiedzi oparte na sześciostopniowej skali. Pytania są pogrupowane w ww. 5 obszarów.

1 Amy Grack Nelson, PhD, Youth Teamwork Skills Survey: Manual and Survey, Science Museum of Minnesota, 2018

Zastosowanie testu

Niniejszy test stanowi źródło informacji nt. stanu początkowego i końcowego indywidualnej percepcji ucznia dotyczącej jego kompetencji współpracy i komunikacji w zespole. Pamiętaj, aby test przeprowadzić w trakcie lekcji wprowadzającej (lekcja 0) oraz lekcji podsumowującej zajęcia (lekcja 6), aby najpełniej zbadać efekt przeprowadzonych zajęć.

Co istotne, test może być wykorzystany do pomiaru samooceny kompetencji w ramach innych zajęć prowadzonych w Twojej szkole, które można zaliczyć do kategorii STEM (ang. science, technology, engineering, mathematics - poza informatyką mogą to być zatem: matematyka, fizyka, chemia czy biologia) - zachęcamy, abyś promował jego stosowanie przez innych nauczycieli, którym zależy na: wiedzy nt. subiektywnej oceny uczniów posiadanych już kompetencji lub pomiarze skuteczności prowadzonych przez nich projektów opartych na pracy zespołowej. Co ważne, test poza naszymi zajęciami można przeprowadzać modułowo, tj. jeśli dany nauczyciel chce zmierzyć poziom kompetencji np. w ramach 3 z 5 ww. obszarów (które uznaje za istotne z punktu widzenia jego zajęć), to z łatwością może to zrobić, bez uszczerbku dla poprawności wyników - trafność wewnętrzna testu została zweryfikowana przez naukowców z SMM. Należy jednak zawsze pamiętać o pozostawieniu scenariusza pracy zespołowej, który rozpoczyna test, oraz zachowaniu kolejności badanych obszarów.

Narzędzie to może być również stosowane do oceny kompetencji młodzieży biorącej udział w zajęciach pozalekcyjnych z zakresu STEM (kółka zainteresowań, kursy weekendowe, szkoły letnie etc.).

Test przeznaczony jest dla młodzieży w wieku od lat 12 do 18.

Założenia organizacyjne

Czas, który powinieneś przeznaczyć na wypełnienie testu przez uczniów, to nie więcej niż 15 minut (zakładamy, że przeciętnie będzie to 5 do 10 minut). Zachęcamy Cię, abyś korzystał z narzędzi cyfrowych wspomagających ankietowanie - na potrzeby niniejszych zajęć udostępniamy Ci do pobrania szablon przygotowany w MsForms, który możesz zduplikować i udostępnić uczniom w ramach posiadanego konta Microsoft (uczniowie nie muszą posiadać takiego konta, formularz testu będzie dostępny w przeglądarce). Dzięki

narzędziom online będziesz mógł na żywo podejrzeć, czy wszyscy uczniowie wypełnili już test.

TEST UMIEJĘTNOŚCI WSPÓŁPRACY I KOMUNIKACJI W ZESPOLE

Przeczytaj najpierw poniższy scenariusz dotyczący pracy w zespole. Pamiętaj o nim odpowiadając na pytania w ankiecie.

Scenariusz pracy zespołowej

Wyobraź sobie, że bierzesz udział w pewnym projekcie i właśnie zostałeś przydzielony/na do zespołu, pracującego nad zadaniem. W zespole jest jeszcze troje młodych ludzi. W sumie jest was czworo, dwie dziewczyny i dwóch chłopaków. Wszyscy chodzicie do szkół średnich. Osoby z zespołu spotkałeś dzisiaj po raz pierwszy w życiu.

Zanim zaczęliście wspólną pracę wszyscy uczestnicy projektu przedstawili się imieniem i nazwiskiem oraz podali po pięć ciekawych faktów na swój temat.

Następnie Twój zespół zapoznał się ze szczegółami zadania i upewniliście się, że wszyscy w zespole rozumieją, co mają zrobić. Następnie członkowie zespołu podzielili się ze sobą informacją o tym, jaką posiadają wiedzę z dziedziny, której dotyczy zadanie. Jako zespół stwierdzacie, że mimo to potrzebujecie dowiedzieć się więcej na temat, który jest przedmiotem zadania.

Jako członkowie zespołu podzieliliście się pracą przy wyszukiwaniu informacji online, w książkach i czasopismach oraz wśród informacji dostarczonych przez organizatorów projektu. Po pewnym czasie spotykacie się ponownie, aby podzielić się zgromadzoną wiedzą. Następnie w zespole rozpoczyna się dyskusja i dzielenie pomysłami dotyczącymi tego, co może być potrzebne do ukończenia projektu. Decydujecie, jakie prace należy wykonać oraz kto z zespołu będzie je wykonywał, a następnie zabieracie się do pracy. Wykonujecie swoją pracę zarówno razem, jak i w pojedynkę. Osiągnięcie celu zespołu zależy od zaangażowania i wkładu wszystkich jego członków, tak więc jesteście ze sobą w bliskim kontakcie, aby być pewnym, że znajdujecie się na właściwej ścieżce.

W poniższej ankiecie zostaniesz poproszony o wyobrażenie sobie, że jesteś członkiem wyżej opisanego zespołu i masz wykonać określone czynności. Odpowiedz na pytania ankiety otwarcie i szczerze. Nie ma dobrych ani złych odpowiedzi na te pytania i nie jesteś za nie oceniany.

Poniżej są opisane sytuacje i są zadane pytania. Wyobraź sobie siebie w tych sytuacjach jako członka opisanego wyżej zespołu i odpowiedz na pytanie , jak dobrze lub niedobrze pójdzie ci odnalezienie się w opisanej sytuacji, jak komfortowo lub niekomfortowo będziesz ją odbierał oraz jakie jest prawdopodobieństwo tego, że zrobisz to co jest opisane w odpowiedzi, będąc członkiem tego zespołu. Pamiętaj, że zespół to ty i trójka innych młodych ludzi.

Pomyśl o sytuacji, w której dzielisz się wyszukanymi informacjami na temat wspólnego zadania, o których żaden/żadna z Twoich koleżanek/kolegów dotąd nie wspomniał/a.

1. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

2. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐ raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

3. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

A teraz wyobraź sobie, że dzielisz się swoimi pomysłami z trójką swoich koleżanek i kolegów z zespołu.

Pomyśl o sytuacji, w której wyjaśniasz swój pomysł zespołowi.

4. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

5. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐ raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

6. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

Pomyśl o sytuacji, w której pytasz swoje koleżanki i kolegów w zespole, czy zrozumieli Twój pomysł.

7. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

8. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐ raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

9. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

Pomyśl o sytuacji, w której zachęcasz swoje koleżanki i kolegów z zespołu do zadawania Tobie pytań dotyczących twojego pomysłu, aby się upewnić, że dobrze go zrozumieli.

10. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

11. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐ raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

12. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

Pomyśl o sytuacji, w której dzielisz się pomysłem sądząc, że może on się nie spodobać Twoim koleżankom i kolegom.

13. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

14. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

15. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

Pomyśl o sytuacji, w której przedstawiasz pomysł, który różni się od pomysłu, o którym właśnie skończyliście dyskutować.

16. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

17. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐ raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

18. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

Teraz wyobraź sobie, że Twoje koleżanki i koledzy dzielą się swoimi pomysłami.

Pomyśl o sytuacji, w której prosisz koleżankę/kolegę o wyjaśnienie ich pomysłu w inny sposób, żebyś lepiej mógł go zrozumieć.

19. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

20. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐ raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

21. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

Pomyśl o sytuacji, w której prosisz koleżankę/kolegę o powtórzenie, bo nie jesteś pewien, czy zrozumiałeś właściwie jej/jego pomysł.

22. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?

☐ bardzo źle

☐ źle

☐ raczej źle

☐ umiarkowanie dobrze

☐ dobrze

☐ bardzo dobrze

23. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?

☐ bardzo niekomfortowo

☐ niekomfortowo

☐ raczej niekomfortowo

☐ umiarkowanie komfortowo

☐ komfortowo

☐ bardzo komfortowo

24. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?

☐ bardzo mało prawdopodobne

☐ mało prawdopodobne

☐ niezbyt prawdopodobne

☐ umiarkowanie prawdopodobne

☐ prawdopodobne

☐ bardzo prawdopodobne

Kolejne pytania dotyczą sytuacji, które mogłyby się wydarzyć podczas wspólnej pracy waszego zespołu.

Pomyśl o tym, jak łatwo lub jak trudno byłoby Tobie zrobić to, co jest opisane poniżej.

25.Słuchaj uważnie koleżanki/kolegi z zespołu, która/y dzieli się swoim pomysłem, zamiast koncentrować się na tym, co chcesz powiedzieć zespołowi o Twoim pomyśle.

☐ bardzo trudno

☐ trudno

☐ niezbyt trudno

☐ umiarkowanie łatwo

☐ łatwo

☐ bardzo łatwo

26. Utrzymuj koncentrację na rozmowie z Twoim zespołem, nie odpływaj myślami.

☐ bardzo trudno

☐ trudno

☐ niezbyt trudno

☐ umiarkowanie łatwo

☐ łatwo

☐ bardzo łatwo

27. W pełni skoncentruj się na tym, co mówi koleżanka/kolega z zespołu, zamiast myśleć o tym, co za chwilę sam powiesz zespołowi.

☐ bardzo trudno

☐ trudno

☐ niezbyt trudno

☐ umiarkowanie łatwo

☐ łatwo

☐ bardzo łatwo

28. Koncentruj się na tym, co mówi koleżanka/kolega z zespołu, podczas, gdy sam wolałbyś pracować nad swoją częścią zadania zespołowego.

☐ bardzo trudno

☐ trudno

☐ niezbyt trudno

☐ umiarkowanie łatwo

☐ łatwo

☐ bardzo łatwo

ARKUSZ Z KLUCZEM PUNKTACJI

Ogólne wskazówki i sugestie

Dla każdego obszaru, opisanego we wprowadzeniu do niniejszego narzędzia badawczego:

1) numerom przypisanym odpowiedziom przypisz identyczną punktację (np. jeśli "bardzo źle" to odpowiedź nr 1, to do tej odpowiedzi przypisz wynik "1 punkt" etc.)

2) zsumuj punkty dla danego obszaru (6 pytań to 6 wyników do zsumowania etc.), a następnie podziel przez liczbę pytań w danym obszarze, by uzyskać średni wyniki dla obszaru

Następnie dodaj do siebie uzyskane średnie, aby uzyskać wynik końcowy; zachowaj jednak wyniki cząstkowe, gdyż potencjalnie stanowią one cenną informację zwrotną dla uczniów, jak również stanowią źródło wiedzy nt. obszarów, w których uczniowie dostrzegają własne deficyty (może to być wskazówka dla wzmocnienia pewnych akcentów w dalszej pracy zespołowej - zarówno na lekcjach informatyki, jak i pozostałych przedmiotów).

Możesz za pomocą testu ocenić również skuteczność zajęć dla całej klasy / grupy - uśrednij wówczas wyniki uzyskane przez uczniów, sumując ich wyniki indywidualne (zarówno w poszczególnych obszarach, jak i ogółem), a następnie dzieląc je przez liczbę uczniów uczestniczących w zajęciach. Jeśli uznasz to za istotne, możesz agregować wyniki wg określonych cech uczniów (np. płci lub wieku) - pozwoli Ci to ocenić, w jaki sposób zajęcia dopasowane są do tych podgrup i jakie różnice występują między nimi, jeśli chodzi o samoocenę kompetencji współpracy i komunikacji w zespole.

Pamiętaj: zachowaj indywidualne wyniki dla siebie. Jeśli uznasz, że informacja o wyniku może być cenną informacją zwrotną dla ucznia, to oczywiście możesz przekazać uczniowi jego wynik (po zakończeniu testowania). Korzystaj za to z uogólnionych wyników dla grupy uczniów (np. klasy), aby lepiej projektować kolejne zajęcia w zespołach.

Sposób obliczania punktacji dla pięciu obszarów

pytania związane z obszarem 1 - dzielenie się pomysłami i informacją

Pomyśl o sytuacji, w której dzielisz się wyszukanymi informacjami na temat wspólnego zadania, o których żaden/żadna z Twoich koleżanek/kolegów dotąd nie wspomniał/a.

• 1. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 2. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 3. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

Pomyśl o sytuacji, w której wyjaśniasz swój pomysł zespołowi.

• 4. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 5. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 6. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

zsumuj punkty dla obszaru 1 (dla pytań 1-6) i podziel przez 6

pytania związane z obszarem 2 - precyzja przekazu

Pomyśl o sytuacji, w której pytasz swoje koleżanki i kolegów w zespole, czy zrozumieli Twój pomysł.

• 7. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 8. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 9. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

Pomyśl o sytuacji, w której zachęcasz swoje koleżanki i kolegów z zespołu do zadawania Tobie pytań dotyczących twojego pomysłu, aby się upewnić, że dobrze go zrozumieli.

• 10. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 11. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 12. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

zsumuj punkty dla obszaru 2 (dla pytań 7-12) i podziel przez 6 pytania

związane z obszarem 3 - asertywność

Pomyśl o sytuacji, w której dzielisz się pomysłem sądząc, że może on się nie spodobać Twoim koleżankom i kolegom.

• 13. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 14. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 15. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

Pomyśl o sytuacji, w której przedstawiasz pomysł, który różni się od pomysłu, o którym właśnie skończyliście dyskutować.

• 16. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 17. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 18. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

zsumuj punkty dla obszaru 3 (dla pytań 13-18) i podziel przez 6

pytania związane z obszarem 4 - otwartość i uważność w zadawaniu pytań

Pomyśl o sytuacji, w której prosisz koleżankę/kolegę o wyjaśnienie ich pomysłu w inny sposób, żebyś lepiej mógł go zrozumieć.

• 19. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 20. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 21. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

Pomyśl o sytuacji, w której prosisz koleżankę/kolegę o powtórzenie, bo nie jesteś pewien, czy zrozumiałeś właściwie jej/jego pomysł.

• 22. Jak dobrze lub źle byś sobie z tym radził w swoim zespole?	
bardzo źle	1 punkt
źle	2 punkty
raczej źle	3 punkty
umiarkowanie dobrze	4 punkty
dobrze	5 punktów
bardzo dobrze	6 punktów

• 23. Jak komfortowo lub niekomfortowo czułbyś się robiąc to w swoim zespole?	
bardzo niekomfortowo	1 punkt
niekomfortowo	2 punkty
niezbyt komfortowo	3 punkty
umiarkowanie komfortowo	4 punkty
komfortowo	5 punktów
bardzo komfortowo	6 punktów

• 24. Jak bardzo prawdopodobne lub mało prawdopodobne byłoby, abyś zrobił to w swoim zespole?	
bardzo mało prawdopodobne	1 punkt
mało prawdopodobne	2 punkty
niezbyt prawdopodobne	3 punkty
umiarkowanie prawdopodobne	4 punkty
prawdopodobne	5 punktów
bardzo prawdopodobne	6 punktów

zsumuj punkty dla obszaru 4 (dla pytań 19-24) i podziel przez 6

pytania związane z obszarem 5 - koncentracja i uważne słuchanie

Pomyśl o tym, jak łatwo lub jak trudno byłoby Tobie zrobić to, co jest opisane poniżej.

• 25. Słuchaj uważnie koleżanki/kolegi z zespołu, która/y dzieli się swoim pomysłem, zamiast koncentrować się na tym, co chcesz powiedzieć zespołowi o Twoim pomyśle.	
bardzo trudno	1 punkt
trudno	2 punkty
niezbyt trudno	3 punkty
umiarkowanie łatwo	4 punkty
łatwo	5 punktów
bardzo łatwo	6 punktów

• 26. Utrzymuj koncentrację na rozmowie z Twoim zespołem, nie odpływaj myślami.	
bardzo trudno	1 punkt
trudno	2 punkty
niezbyt trudno	3 punkty
umiarkowanie łatwo	4 punkty
łatwo	5 punktów
bardzo łatwo	6 punktów

• 27. W pełni skoncentruj się na tym, co mówi koleżanka/kolega z zespołu, zamiast myśleć o tym, co za chwilę sam powiesz zespołowi.	
bardzo trudno	1 punkt
trudno	2 punkty
niezbyt trudno	3 punkty
umiarkowanie łatwo	4 punkty
łatwo	5 punktów
bardzo łatwo	6 punktów

• 28. Koncentruj się na tym, co mówi koleżanka/kolega z zespołu, podczas, gdy sam wolałbyś pracować nad swoją częścią zadania zespołowego.	
bardzo trudno	1 punkt
trudno	2 punkty
niezbyt trudno	3 punkty
umiarkowanie łatwo	4 punkty
łatwo	5 punktów
bardzo łatwo	6 punktów

zsumuj punkty dla obszaru 5 (dla pytań 25-28) i podziel przez 4

Kody QR do filmów dla uczniów - arkusze do wydrukowania

Filmy dotyczące umiejętności współpracy i komunikacji w zespole

Kod do playlisty z filmami

https://diode.zone/w/p/7gvypgZXZ8NgKA1T4DTc91

Lekcja 1

Kod do playlisty z filmami

https://diode.zone/w/p/h6avLb6f7bDjhFdvKMR5JH

Praca samodzielna po lekcji 1

Kod do playlisty z filmami

https://diode.zone/w/p/gC8EgZbLap8xrrAXvRUgfs

Lekcja 2

Kod do playlisty z filmami

https://diode.zone/w/p/egaBcYgeEP8PCVPnzt3fdQ

Praca samodzielna po lekcji 2

Kod do playlisty z filmami

https://diode.zone/w/p/vK9AbX1KJqAdKmbMrueEy8

Lekcja 3

Kod do playlisty z filmami

https://diode.zone/w/p/bVMsEq4JYbKnRMuBTcMyJR

Praca samodzielna po lekcji 3

Kod do playlisty z filmami

https://diode.zone/w/p/n9V9XyMLuRR7GTiWqSubbM

Lekcja 4

Kod do playlisty z filmami

https://diode.zone/w/p/n5XuNCVLZaQFbrAHXNrHFQ

Praca samodzielna po lekcji 4

Kod do playlisty z filmami

https://diode.zone/w/p/xxEQjT3cTqDNDgbutjXPu1

Lekcja 5

Kod do playlisty z filmami

https://diode.zone/w/p/9BWGTTMDTZ8So2NViSw43a