

WYMAGANIA EDUKACYJNE I KRYTERIA OCENIANIA Z INFORMATYKI

dla klasy 8 Szkoły Podstawowej | Rok szkolny 2026/2027

I. ZASADY OGÓLNE I FORMY OCENIANIA

1. Główny nacisk: Ze względu na praktyczny charakter przedmiotu oceniani podlega przede wszystkim wiedza oraz umiejętności praktyczne prezentowane przez ucznia podczas pracy przy komputerze.

2. Formy aktywności podlegające ocenie:

- Praca bieżąca na lekcji: wykonywanie ćwiczeń, zadań komputerowych oraz bieżąca aktywność i zaangażowanie.
- Komputerowe zadania praktyczne (prace praktyczne): praktyczne sprawdziany umiejętności realizowane przy komputerze (wykonanie serii poleceń, obsługa programów, analiza wyników) połączone z weryfikacją podstawowych pojęć (np. testy, zadania na uzupełnianie brakujących komend, kodu lub nazw).
- Projekty indywidualne oraz grupowe: dłuższe prace tworzone przy użyciu poznanych narzędzi cyfrowych (np. prezentacje, skrypty/programy, prace graficzne).
- Przestrzeganie zasad BHP, regulaminu pracowni oraz cyberbezpieczeństwa i prawa autorskiego.

3. Materiały pomocnicze (ZPE): Uczniowie mogą wspierać swoją naukę materiałami udostępnianymi na Zintegrowanej Platformie Edukacyjnej (zpe.gov.pl). Korzystanie z platformy ma charakter wspomagający i uzupełniający wiedzę.

4. Regulamin pracowni: Uczeń ma obowiązek przestrzegać zasad bezpiecznej pracy z komputerem. Samowolne zmienianie konfiguracji sprzętu, oprogramowania lub niestosowanie się do BHP może wpływać na ocenę częściową z aktywności oraz ocenę z zachowania.

II. OGÓLNE KRYTERIA NA POSZCZEGÓLNE OCENY

Sposób łączenia wymagań na oceny: Ocena wyższa obejmuje również wszystkie umiejętności z ocen niższych. Oznacza to, że aby otrzymać np. ocenę dobrą (4), uczeń musi opanować nie tylko materiał przypisany do tej oceny, ale również w pełni spełniać wymagania na ocenę dopuszczającą (2) i dostateczną (3).

Ocena celująca (6): Uczeń sprawnie i samodzielnie posługuje się poznanym oprogramowaniem, rozwiązuje zadania o wysokim stopniu złożoności, proponuje autorskie/kreatywne rozwiązania, sprawnie i bezbłędnie rozwiązuje złożone zadania praktyczne oraz projektowe, a także chętnie pomaga innym.

Ocena bardzo dobra (5): Uczeń w pełni opanował umiejętności przewidziane w programie danej klasy, pracuje sprawnie i samodzielnie wykonuje komputerowe zadania praktyczne oraz wykazuje wysoką dbałość o estetykę i poprawność swoich prac.

Ocena dobra (4): Uczeń opanował większość umiejętności z programu, poprawnie wykonuje zadania praktyczne na komputerze (z ewentualną drobną pomocą nauczyciela), sprawnie korzysta z komputera i zasobów sieci.

Ocena dostateczna (3): Uczeń opanował podstawowe umiejętności pozwalające na rozwiązywanie prostych zadań, zadania praktyczne przy komputerze wykonuje w stopniu wystarczającym, czasem wymaga wsparcia przy bardziej złożonych operacjach.

Ocena dopuszczająca (2): Uczeń opanował elementarne umiejętności umożliwiające mu pracę z komputerem przy stałej pomocy nauczyciela, potrafi wykonać podstawowe polecenia podczas zadań praktycznych i zna zasady BHP.

Ocena niedostateczna (1): Uczeń nie opanował elementarnych umiejętności, nie potrafi wykonać prostych zadań nawet przy wsparciu nauczyciela, wykazuje lekceważący stosunek do przedmiotu lub łamie zasady BHP.

III. ZASADY POPRAWIANIA OCEN I KLASYFIKACJI ŚRÓDROCZNEJ ORAZ ROCZNEJ

1. Poprawa ocen częściowych:

- Uczeń ma prawo do jednorazowej poprawy oceny z komputerowych zadań praktycznych (sprawdzianów umiejętności) oraz projektów w terminie 14 dni od jej ogłoszenia (lub w terminie uzgodnionym z nauczycielem).
- W przypadku poprawy w dzienniku elektronicznym wpisuje się ocenę uzyskaną z poprawy obok oceny pierwotnej (przy ustalaniu oceny klasyfikacyjnej uwzględnia się ocenę korzystniejszą dla ucznia).
- Oceny za bieżącą pracę na lekcji oraz aktywność nie podlegają ponownej poprawie.
- W przypadku usprawiedliwionej nieobecności na zapowiadzanych zadaniach praktycznych, uczeń ma obowiązek wykonać je w dodatkowym terminie ustalonym z nauczycielem.

2. Ustalanie oceny śródrocznej i rocznej:

- Ocena śródroczna i roczna nie jest średnią arytmetyczną ocen częściowych.
- Najważniejszy wpływ na ocenę klasyfikacyjną mają oceny uzyskane z komputerowych zadań praktycznych oraz projektów indywidualnych/grupowych.

IV. SZCZEGÓŁOWE WYMAGANIA EDUKACYJNE – KLASA 8

Wymagania zostały zorganizowane według działów tematycznych na podstawie podręcznika Nowej Ery „Lubię to!” (Python) dla klasy 8. Ocena wyższa wymaga opanowania umiejętności przypisanych do ocen niższych.

Ocena	Wymagania edukacyjne z podziałem na działy i programy
Dopuszczająca (2)	Arkusz kalkulacyjny (MS Excel / Google Sheets): • Omawia zastosowanie i budowę arkusza; określa adres komórki i wprowadza dane różnego rodzaju. • Formatuje wyrównanie tekstu oraz wygląd czcionki w komórkach. • Rozumie różnice między adresowaniem względnym, bezwzględnym i mieszanym. • Wstawia wykres do arkusza i tworzy prostą kalkulację wydatków. Programowanie w języku Python i algorytmika: • Definiuje pojęcia: algorytm, program, programowanie; podaje sposoby przedstawiania algorytmów. • Tłumaczy przeznaczenie zmiennych w programie; pisze proste programy w trybie skryptowym. • Wyjaśnia działanie operatora modulo (%) i algorytm badania podzielności liczb naturalnych. • Wyjaśnia potrzebę wyszukiwania i porządkowania danych w zbiorze. • Sprawdza działanie programów wyszukujących oraz sortujących dla podanych danych. Praca w projekcie zespołowym: • Bierze udział w tworzeniu dokumentacji szkolnego wydarzenia (np. imprezy sportowej) lub tworzeniu gry, wykonując przydzielone proste zadania i testując grę na różnych etapach.
Dostateczna (3)	Arkusz kalkulacyjny (MS Excel / Google Sheets): • Określa zasady wprowadzania danych, dodaje i usuwa wiersze oraz kolumny w tabeli. • Stosuje podstawowe funkcje (SUMA, ŚREDNIA) – wpisuje je ręcznie oraz korzysta z kreatora.

- Omawia i modyfikuje elementy wykresu oraz dobiera odpowiedni wykres do rodzaju danych.
- Zapisuje w tabeli dane z prostych doświadczeń i przedstawia je na wykresie.

Programowanie w języku Python i algorytmika:

- Wyjaśnia różnicę między trybem interaktywnym a skryptowym, kodem źródłowym a wynikowym oraz rolę środowiska IDE.
- Używa `print()` do wyświetlania tekstu; wykonuje obliczenia w Pythonie i omawia operatory arytmetyczne.
- Zapisuje algorytm badania podzielności w postaci listy kroków; stosuje pętlę `while` w programach.
- Zapisuje algorytm wyszukiwania w zbiorze nieuporządkowanym (w tym elementu `min/max`) oraz algorytm sortowania przez wybieranie.
- Wyjaśnia działanie pętli zagnieżdżonych.

Praca w projekcie zespołowym:

- Wprowadza dane do tabel projektowych, aktywnie uczestniczy w pracach zespołu i bierze udział w tworzeniu koncepcji gry.

Dobra (4)

Arkusz kalkulacyjny (MS Excel / Google Sheets):

- Tworzy proste formuły, kopiuje je przy użyciu adresowania względnego.
- Wykorzystuje funkcję JEŻELI do algorytmów z warunkami i ustawia odpowiednie formaty danych w komórkach.
- Tworzy wykres dla więcej niż jednej serii danych; sortuje oraz filtruje dane w arkuszu.

Programowanie w języku Python i algorytmika:

- Zna specyfikację problemu, etapy rozwiązywania problemów i powstawania programu.
- Stosuje instrukcje warunkowe (`if`, `if...else`), pętlę iteracyjną `for`, listy oraz operatory logiczne.
- Omawia algorytm Euklidesa, algorytm wyodrębniania cyfr liczby oraz implementuje wyszukiwanie w zbiorze nieuporządkowanym.
- Omawia funkcje w kodzie źródłowym sortowania przez wybieranie.

Praca w projekcie zespołowym:

- Przygotowuje dokumentację projektową (obliczenia, tabele, wykresy), programuje wybrane funkcje gry i opracowuje jej opis.

Bardzo dobra (5)

Arkusz kalkulacyjny (MS Excel / Google Sheets):

- Tworzy i kopiuje skomplikowane formuły przy użyciu adresowania bezwzględnego (\$) i mieszanego.
- Korzysta z biblioteki funkcji; tworzy rozbudowane wykresy dla wielu serii danych.
- Tworzy prosty model symulacyjny (np. rzut kostką) i stosuje filtry niestandardowe.

Programowanie w języku Python i algorytmika:

- Pisz programy matematyczne (złożone warunki, pętle, listy, funkcje zwracające/niezwracające wartości) i czyta kod źródłowy.
- Wyjaśnia różnicę między pętlą `while` a `for`; pisze programy obliczające NWD (algorytm Euklidesa) oraz wyodrębniające cyfry.
- Samodzielnie zapisuje i implementuje algorytm sortowania przez wybieranie oraz wprowadza w nim modyfikacje.

- Implementuje i optymalizuje kod źródłowy gry na podstawie założeń.

Praca w projekcie zespołowym:

- Przygotowuje pełne zestawienia projektowe, drukuje wyniki i sprawnie współpracuje w grupie.

Celująca (6)

- Tworzy rozbudowane, profesjonalne arkusze kalkulacyjne stosujące zaawansowane funkcje i modele do analizy danych z innych przedmiotów.
 - Samodzielnie modyfikuje, optymalizuje i pisze rozbudowane programy w Pythonie z użyciem zaawansowanych algorytmów (np. NWW, optymalizacja sortowania i wyszukiwania).
 - Projektuje i buduje własne moduły oraz funkcje w programie głównym Python.
 - Tworzy autorskie, rozbudowane gry komputerowe w Pythonie oraz wykazuje się cechami lidera w projektach zespołowych, wspierając innych uczniów.
-