# Wymagania edukacyjne z informatyki w klasie 2 LO – zakres rozszerzony

**Ocenę celującą** otrzymuje uczeń, który spełnia kryteria ocen niższych, a ponadto:

- charakteryzuje skomplikowane sytuacje algorytmiczne, proponuje optymalne rozwiązanie sytuacji problemowej z zastosowaniem złożonych struktur danych i biblioteki STL języka C++,
- pisze programy o wysokim stopniu trudności: z olimpiad przedmiotowych, konkursów informatycznych lub oznaczone trzema gwiazdkami w podręczniku,
- wyszukuje palindromy lub anagramy w plikach tekstowych,
- tworzy palindromy z napisów, dopisując minimalną liczbę znaków,
- pisze program rozkładający liczbę złożoną na dwie liczby pierwsze (hipoteza Goldbacha),
- implementuje w języku C++ algorytm Euklidesa, stosując iterację i rekurencję,
- pisze programy szyfrujące i deszyfrujące z wykorzystaniem zaawansowanych szyfrów (np. permutacyjny lub Vigenere'a) i różnych kluczy,
- implementuje w języku C++ algorytm wyszukiwania binarnego w wersji rekurencyjnej,
- pisze programy sortujące dane różnego typu w plikach tekstowych (liczby, napisy, pary),
- stosuje zaawansowane algorytmy i struktury danych do wyszukiwania spójnych podciągów,
- stosuje zaawansowane algorytmy wyszukiwania, np. najlepszego wyboru (trwałych par), stosując rekurencję,
- pisze programy obliczające liczbę operacji przenoszenia krążków w problemie wież Hanoi, stosując iterację i rekurencję,
- stosuje w programach algorytmy sortowania inne niż omawiane na lekcjach (np. heapsort),
- bierze udział w olimpiadach i konkursach, zajmując punktowane miejsca,
- w projektach zespołowych przyjmuje rolę lidera.

**Ocenę bardzo dobrą** otrzymuje uczeń, który spełnia kryteria ocen niższych, a ponadto:

- charakteryzuje sytuacje algorytmiczne, proponuje sposoby ich rozwiązania,
- pisze programy o podwyższonym stopniu trudności: oznaczone trzema gwiazdkami w podręczniku,
- optymalizuje rozwiązania,
- stosuje zaawansowane funkcje środowiska i języka programowania (np. z biblioteki STL),
- dobiera struktury danych i metody do rodzaju problemu,
- pisze programy konwertujące liczby między różnymi systemami pozycyjnymi,
- w programach wykonujących działania na liczbach w różnych systemach pozycyjnych wykorzystuje bibliotekę string i strukturalne typy danych,
- wykorzystuje rozwinięcie binarne liczby dziesiętnej w algorytmie szybkiego podnoszenia do potęgi,
- wykonuje operacje arytmetyczne na liczbach w różnych systemach, implementuje je w języku C++,
- stosuje różne sposoby przekazywania parametrów do funkcji, uzasadnia ich użycie,
- pisze funkcje typu logicznego, np. sprawdzającą, czy napis jest palindromem,
- sprawdza, czy napisy są anagramami, stosując sortowanie lub zliczanie znaków,
- przy testowaniu liczby na pierwszość stosuje funkcję typu logicznego,
- wyszukuje liczby bliźniacze,
- wykorzystuje algorytm Euklidesa do działań na ułamkach, stosując struktury lub pary (typ pair),

- szyfruje dane wczytane z pliku z uwzględnieniem polskich znaków diakrytycznych,
- pisze program odczytujący informację ukrytą za pomocą szyfru Cezara z wykorzystaniem analizy częstości znaków w tekście,
- pisze program wyszukujący jednocześnie minimum i maksimum w zbiorze z wykorzystaniem metody „dziel i zwyciężaj” oraz podaje wzór na liczbę wykonywanych operacji,
- szacuje złożoność obliczeniową programów sortujących, modyfikuje funkcje sortujące, zmieniając porządek sortowania,
- wykorzystuje poznane algorytmy do rozwiązywania problemów nieomawianych na lekcjach,
- optymalizuje program realizujący algorytm sita Eratostenesa i szacuje jego złożoność czasową,
- wyszukuje spójne podciągi w plikach tekstowych, stosując optymalne algorytmy (w tym programowanie dynamiczne), wyjaśnia ich działanie,
- pisze programy wyszukujące lidera i idola w zbiorze, optymalizuje je, szacuje złożoność czasową,
- pisze programy obliczające liczbę operacji przenoszenia krążków w problemie wież Hanoi, stosując iterację i rekurencję,
- do implementacji rozszerzonego algorytmu Euklidesa stosuje zarówno iterację, jak i rekurencję,
- stosuje metody zachłanną i dynamiczną w problemach kasjera, harmonogramu wykorzystania sali, pakowania plecaka i wyszukiwania drogi, wskazuje wady i zalety obu metod, szacuje złożoność czasową,
- aktywnie uczestniczy w realizacji projektu zespołowego na wszystkich jego etapach, prezentuje efekty wspólnej pracy.

**Ocenę dobrą** otrzymuje uczeń, który spełnia kryteria ocen niższych, a ponadto:

- określa specyfikację algorytmu (dane, wynik),
- pisze programy o różnym stopniu trudności, szacuje ich efektywność,
- przedstawia omawiane algorytmy w postaci opisu słownego, listy kroków, schematu blokowego, pseudokodu,
- dobiera typy danych do realizacji problemu,
- stosuje zmienne typu unsigned w tworzonych programach,
- pisze programy konwertujące liczby między systemem dziesiętnym i binarnym,
- implementuje w języku C++ algorytmy wykonujące działania arytmetyczne na liczbach w różnych systemach,
- w algorytmach zamiany wykorzystuje zależności między systemami binarnym, ósemkowym i heksadecymalnym,
- omawia sposób reprezentacji obrazów w komputerze, korzystając z takich pojęć jak: piksel, model RGB, kanał alfa,
- wyjaśnia, na czym polega digitalizacja (dyskretyzacja) dźwięku,
- wyjaśnia zasadę tworzenia animacji,
- stosuje różne sposoby przekazywania parametrów do funkcji: przez wartość, referencję lub wskaźnik,
- implementuje w języku C++ algorytmy sprawdzające, czy napis jest palindromem,
- pisze programy sprawdzające, czy dwa napisy są anagramami, wykorzystując funkcję sort z biblioteki STL,

- implementuje w języku C++ i optymalizuje algorytm sprawdzający, czy liczba jest pierwsza,
- pisze program rozkładający liczby na czynniki pierwsze,
- stosuje w programach algorytm Euklidesa do obliczenia NWD i NWW,
- wykorzystuje algorytm Euklidesa do działań na ułamkach,
- szyfruje dane wczytane z pliku tekstowego,
- implementuje w języku C++ algorytm zliczania znaków w tekście oraz wyszukujący maksimum z wykorzystaniem tablic,
- stosuje algorytm wyszukiwania binarnego i oszacowuje jego złożoność czasową,
- pisze programy sortujące metodami prostymi z zastosowaniem funkcji typu void,
- stosuje algorytmy sortowania szybkiego i przez scalanie,
- pisze program realizujący algorytm sita Eratostenesa,
- implementuje w języku C++ algorytmy wyszukujące spójne podciągi o różnych cechach,
- stosuje w programach algorytmy wyszukiwania lidera i idola w zbiorze,
- porównuje algorytmy iteracyjne i rekurencyjne (liczbę wykonywanych operacji), szacuje ich złożoność czasową,
- zapisuje w postaci programu rozszerzony algorytm Euklidesa, wyjaśnia jego działanie i zastosowanie,
- stosuje metodę zachłanną w programach – problem kasjera, harmonogram wykorzystania sali, wyszukiwanie drogi, pakowanie plecaka,
- uczestniczy w realizacji projektu zespołowego na wszystkich jego etapach, bierze czynny udział w tworzeniu dokumentacji projektowej.

**Ocenę dostateczną** otrzymuje uczeń, który spełnia kryteria oceny dopuszczającej, a ponadto:

- przedstawia krótkie algorytmy w postaci listy kroków, opisu słownego, pseudokodu, schematu blokowego,
- dodaje liczby binarne,
- konwertuje liczby między pozycyjnymi systemami liczbowymi,
- wykonuje działania arytmetyczne na liczbach w systemach liczbowych o różnych podstawach,
- przedstawia liczby w kodzie U2,
- definiuje pojęcie zdania logicznego, charakteryzuje podstawowe operacje logiczne (koniunkcja, alternatywa, negacja) oraz operatory logiczne,
- charakteryzuje wybrane typy zmiennych służących do zapisu liczb całkowitych w języku C++: short int, int, long int, long long int,
- pisze programy wykonujące działania na liczbach całkowitych,
- korzysta z biblioteki string do operacji na łańcuchach znaków,
- wykonuje operacje na napisach, wykorzystując słowa kluczowe: size, find, substr, erase, toupper, tolower,
- wczytuje napisy ze spacjami, wykorzystując słowo kluczowe getline,
- tworzy algorytmy sprawdzające, czy napis jest palindromem,
- przedstawia w postaci algorytmu problem wyszukiwania anagramów,
- przy pisaniu programów stosuje własne funkcje różnych typów, w tym funkcję typu void,
- wyjaśnia różnicę między parametrami formalnym i aktualnym, a także między zmiennymi lokalną i globalną,
- implementuje w języku C++ algorytm naiwny sprawdzający, czy liczba jest pierwsza,

- implementuje w języku C++ algorytm Euklidesa w wersjach z dzieleniem i odejmowaniem,
- pisze program szyfrujący napis szyfrem Cezara,
- omawia algorytm zliczania znaków w tekście oraz wyszukujący maksimum z wykorzystaniem tablic,
- implementuje w języku C++ algorytmy wyszukiwania liniowego i liniowego z wartownikiem, porównuje ich efektywność,
- przedstawia w postaci listy kroków lub schematu blokowego algorytmy sortowania prostego (bąbelkowe, przez wybieranie) oraz szybkiego i przez scalanie, określa operacje dominujące,
- omawia algorytm sita Eratostenesa,
- przedstawia algorytmy znajdowania spójnych podciągów, wyznaczania najdłuższego z nich oraz podciągu o największej sumie elementów,
- omawia algorytm znajdowania idola i lidera w zbiorze,
- implementuje w języku C++ algorytmy rekurencyjne: obliczanie elementów ciągu Fibonacciego, wartości silni i potęgi,
- omawia rozszerzony algorytm Euklidesa,
- formułuje algorytm wydawania reszty minimalną liczbą monet, harmonogramu wykorzystania sali, pakowania plecaka, znajdowania drogi metodami zachłanną i dynamiczną,
- uczestniczy w realizacji projektu zespołowego, wykonując powierzone mu zadania.

**Ocenę dopuszczającą** otrzymuje uczeń, który:

- definiuje podstawowe pojęcia z algorytmiki i programowania: algorytm, program, warunek, iteracja, rekurencja,
- wymienia sposoby reprezentacji algorytmów,
- korzysta ze środowiska programistycznego: pisze w nim kod, kompiluje i uruchamia program, odczytuje i zapisuje pliki,
- pisze programy o niewielkim stopniu trudności,
- omawia pojęcia: złożoność obliczeniowa algorytmu, algorytm naiwny, algorytm optymalny, złożoność pesymistyczna, złożoność oczekiwana (średnia),
- korzysta z podstawowych funkcji języka: operacji wejścia i wyjścia, instrukcji warunkowych i iteracyjnych, gotowych funkcji bibliotecznych,
- wymienia podstawowe typy danych, operacje arytmetyczne i logiczne,
- w pisanych programach korzysta ze strukturalnych typów danych: napisów, struktur, tablic,
- definiuje pojęcie systemów liczbowych,
- wyjaśnia, czym jest tablica kodów ASCII,
- wymienia systemy liczbowe używane w informatyce,
- konwertuje liczby między systemami binarnym i decymalnym,
- dodaje pisemnie liczby binarne,
- wyjaśnia, czym są palindrom i anagram, podaje przykłady,
- podaje definicje liczby pierwszej i liczby złożonej,
- implementuje w języku C++ algorytm zliczający dzielniki danej liczby,
- omawia geometryczną interpretację algorytmu Euklidesa,
- definiuje pojęcia: kryptologia, kryptografia, kryptoanaliza, tekst jawny, klucz, szyfrogram,
- rozróżnia szyfry podstawieniowe i przestawieniowe,

- omawia szyfr Cezara jako przykład szyfru podstawieniowego i szyfr kolumnowy jako przykład szyfru przestawieniowego,
- wyjaśnia, na czym polega łamanie szyfru,
- omawia algorytm zliczania znaków w tekście,
- wyjaśnia, na czym polega metoda „dziel i zwyciężaj”,
- wczytuje dane z pliku tekstowego, zapisuje wyniki w pliku,
- omawia algorytmy wyszukiwania liczby w zbiorach uporządkowanym i nieuporządkowanym,
- stosuje funkcję losującą w tworzonych programach,
- omawia metody sortowania prostego (bąbelkowe, przez wybieranie) oraz szybkiego i przez scalanie na przykładowych danych,
- wypisuje liczby pierwsze z zadanego przedziału, stosując metodę sita Eratostenesa,
- wyszukuje w ciągu liczb spójne podciągi (nierosnący, niemalejący, stały), wskazuje najdłuższe, oblicza ich sumę,
- wskazuje idola i lidera w zbiorze danych,
- definiuje pojęcia iteracji i rekurencji,
- omawia zasadę złotego podziału,
- opisuje rozszerzony algorytm Euklidesa,
- omawia metody zachłanne na przykładzie problemu kasjera, harmonogramu sali, pakowania plecaka i wyszukiwania drogi,
- porównuje metody zachłanną i dynamiczną,
- uczestniczy w realizacji projektu zespołowego, wykonując powierzone mu zadania o niewielkim stopniu trudności.

**Ocenę niedostateczną** otrzymuje uczeń, który:

- nie opanował podstawowych wiadomości i umiejętności, co uniemożliwia zdobywanie dalszej wiedzy,
- nie jest w stanie scharakteryzować podstawowych pojęć (algorytm, warunek, iteracja, rekurencja),
- nie zna prostych algorytmów,
- nie rozwiązuje najprostszych zadań,
- nie bierze czynnego udziału w lekcjach, nie wykonuje zadań, nie pisze programów, nie odrabia prac domowych.