Wymagania edukacyjne niezbędne do otrzymania przez ucznia śródrocznej i rocznej oceny klasyfikacyjnej z informatyki

Klasa 2.

Zakres rozszerzony

Temat lekcji	Zagadnienia	Wymagania edukacyjne na poszczególne oceny				
		Ocena dopuszczająca **Uczeń:**	**Ocena dostateczna** **Uczeń:**	**Ocena dobra** **Uczeń:**	**Ocena bardzo dobra** **Uczeń:**	**Ocena celująca** **Uczeń:**
Wymagania edukacyjne niezbędne do otrzymania przez ucznia śródrocznej oceny klasyfikacyjnej						
Rozdział I. Algorytmy na liczbach całkowitych i tekstach						
Od problemu do programu	• Algorytmy • Podstawy języka C++	— wyjaśnia pojęcie algorytmu — podaje przykłady algorytmów w życiu codziennym — kompiluje zapisany kod źródłowy	— wymienia cechy poprawnego algorytmu — wyjaśnia na przykładzie pojęcie specyfikacji problemu — tworzy algorytm wyznaczania pierwiastka kwadratowego	— zapisuje algorytm Herona w postaci listy kroków — wyjaśnia pojęcia związane z algorytmiką i programowaniem: schemat blokowy, lista kroków, kod źródłowy, kod wynikowy, kompilator, interpreter, słowa kluczowe, funkcje, plik wykonywalny — zapisuje	— znajduje i poprawia błędy w kodzie źródłowym programu — wyjaśnia pojęcie zmiennej i typu zmiennej — wymienia zasady tworzenia kodu źródłowego w wybranym języku programowania — stosuje podstawowe konstrukcje wybranego języka	— tworzy samodzielnie programy, wykorzystując poznane instrukcje wybranego języka programowania — stosuje w swoich programach zagnieżdżone instrukcje warunkowe — pisze programy rozwiązujące zadania matematyczne i fizyczne oraz problemy z

				algorytm w postaci kodu źródłowego	programowania: instrukcje wejścia i wyjścia, operatory arytmetyczne i logiczne oraz instrukcję warunkową — tworzy program sprawdzający warunek trójkąta	napisami
Systemy liczbowe i reprezentacja danych w komputerze	• System binarny • System heksadecymalny	— definiuje pojęcie pozycyjnego systemu liczbowego — wymienia systemy liczbowe stosowane w informatyce	— definiuje pojęcia bit i bajt — dokonuje konwersji między pozycyjnymi systemami liczbowymi, wykorzystując przy tym zależności między systemami binarnym i ósemkowym oraz binarnym i heksadecymalnym	— omawia sposób reprezentowania liczb całkowitych w komputerze — wyjaśnia, czym jest tablica kodów ASCII	— wymienia typy danych służące do zapisu liczb całkowitych (short int, int, long int, long long int, unsigned), stosuje je w pisanych programach — opisuje, jak w komputerze reprezentowane są znaki i napisy (char, string), odwołuje się do znaku w	— omawia działanie operacji logicznych — wykonuje zadania oznaczone trzema gwiazdkami w podręczniku, z arkuszy maturalnych z lat poprzednich lub konkursów i olimpiad informatycznych

					napisie za pomocą indeksu	
Algorytmy zamiany reprezentacji liczb między systemami liczbowymi	• Algorytm zamiany liczby dwójkowej na dziesiętną i odwrotnie	— tworzy programy do konwersji między liczbami w systemach binarnym i decymalnym	— posługuje się środowiskiem programistycznym, strukturami danych oraz językiem programowania w stopniu umożliwiającym implementację omawianych algorytmów	— pisze programy konwertujące liczbę dziesiętną na liczbę w podanym systemie pozycyjnym	— stosuje binarną reprezentację liczby w algorytmie szybkiego podnoszenia do potęgi	— pisze programy zamieniające liczby z systemu decymalnego na system heksadecymalny — pisze z wykorzystaniem algorytmów zamiany: z zadań oznaczonych trzema gwiazdkami w podręczniku, z arkuszy maturalnych, z konkursów i olimpiad informatycznych — do rozwiązania problemu dobiera optymalny algorytm i struktury danych

Czy to jest palindrom?	• Palindromy	— definiuje pojęcie palindromu	— określa, czy dany napis lub liczba są palindromami	— opisuje popularne funkcje oraz metody stosowane dla zmiennych typu string (toupper, tolower, size, substr, erase) — wykonuje operacje na napisach (wczytywanie napisów ze spacjami, sprawdzanie długości napisu, zamiana liter dużych na małe i odwrotnie, porównywanie znaków, znajdowanie oraz usuwanie fragmentów napisów)	— definiuje własne funkcje w języku C++, wyjaśnia celowość ich stosowania, rozróżnia parametry formalne i aktualne — realizuje w języku C++ algorytmy sprawdzające, czy dany napis jest palindromem, oraz wyszukujące palindromy w zdaniach	— optymalizuje algorytmy i ocenia ich efektywność

Czy ta liczba jest pierwsza?	• Liczby złożone i pierwsze • Podzielność liczb	— wymienia podstawowe własności liczb pierwszych	— sprawdza, czy dana liczba jest pierwsza, stosując algorytm naiwny	— rozkłada liczbę złożoną na czynniki pierwsze — wyznacza liczby bliźniacze	— tworzy samodzielnie programy dla poznanych algorytmów	— implementuje algorytmy dotyczące liczb pierwszych — stosuje optymalny algorytm sprawdzający, czy liczba jest pierwsza, wykorzystując funkcję logiczną; uzasadnia jego efektywność — pisze program rozkładający liczbę złożoną na sumę dwóch liczb pierwszych (hipoteza Goldbacha)
Działania na liczbach w systemach innych niż dziesiętny	• Dodawanie, odejmowanie, mnożenie i dzielenie w różnych systemach pozycyjnych	— wykonuje działania arytmetyczne na liczbach w różnych systemach pozycyjnych	— wykonuje obliczenia na dowolnie dużych liczbach, wykorzystując napisy	— wyjaśnia różnicę między operacjami na liczbach o podstawie od 1 do 9 i większej od 10	— stosuje odejmowanie w dzieleniu pisemnym liczb binarnych	— stosuje dodawanie liczby przeciwnej zapisanej w kodzie U2 przy odejmowaniu liczby binarnej

Algorytm Euklidesa i działania na ułamkach	• Największy wspólny dzielnik • Działania na ułamkach zwykłych	— wyjaśnia pojęcia: NWD, NWW	— podaje przykłady zastosowania algorytmu Euklidesa — zapisuje algorytm Euklidesa w postaci listy kroków	— tworzy program pozwalający na dodawanie ułamków — stosuje odpowiednie konstrukcje wybranego języka programowania do implementacji omawianych zagadnień (w tym: funkcję, która nie zwraca wartości)	— tworzy programy realizujące działania na ułamkach	— opisuje algorytm Euklidesa i tworzy realizujący go program w wybranym języku programowania — opisuje różnicę w sprawności dwóch wersji algorytmu Euklidesa: z odejmowaniem i z dzieleniem — poznaje inne zastosowania algorytmu Euklidesa, wykorzystując informacje zawarte w internecie lub innych źródłach

Szyfr Cezara i inne szyfry podstawieniowe	• Kryptografia • Szyfr Cezara • Szyfr kolumnowy	— definiuje pojęcia – kryptologia, kryptografia, kryptoanaliza, informacja jawna, szyfrogram, klucz szyfrowania — rozróżnia szyfry przestawieniowe i podstawieniowe	— wymienia metody łamania klasycznych szyfrów (atak siłowy, analiza częstości)	— implementuje algorytmy szyfrujące metodą kolumnową	— implementuje algorytmy szyfrujące i deszyfrujące metodą Cezara — stosuje pętle zagnieżdżone	— definiuje pojęcia klucz symetryczny i niesymetryczny w algorytmach szyfrowania — omawia i implementuje inne algorytmy szyfrowania (np.: szyfry: Beauforta, skokowy, afiniczny Vigenere'a, algorytm RSA
Wymagania edukacyjne niezbędne do otrzymania przez ucznia rocznej oceny klasyfikacyjnej (obejmują wymagania edukacyjne niezbędne do otrzymania przez ucznia śródrocznej oceny klasyfikacyjnej).						
Rozdział II. Rozwiązywanie problemów z wykorzystaniem struktur danych						
Łamiemy szyfr Cezara	• Metody łamania szyfrów • Zliczanie liter w tekście • Szukanie maksimum w zbiorze	— wyjaśnia, na czym polega łamanie szyfru (kryptoanaliza)	— łamie szyfr Cezara, stosując analizę częstości	— stosuje algorytmy zliczające liczbę wystąpień znaków w tekście z zastosowaniem strukturalnego typu danych –	— pisze program znajdujący maksimum w tablicy i wypisujący jego pozycję (algorytm „dziel i zwyciężaj")	— opisuje różne sposoby łamania szyfrów i implementuje je w języku C++

				tablic		
Poszukujemy liczby	• Wyszukiwanie liczby w zbiorze nieuporządkowanym • Wyszukiwanie liczby w zbiorze uporządkowanym	— omawia algorytm wyszukiwania liniowego	— znajduje wartość w zbiorach uporządkowanym i nieuporządkowanym, stosując odpowiednio algorytmy wyszukiwania liniowego, liniowego z wartownikiem i binarnego	— pisze programy wykorzystujące przekazywanie parametru do funkcji przez wskaźnik i referencję	— stosuje algorytm „dziel i zwyciężaj” do jednoczesnego znajdowania maksimum i minimum w zbiorze	— szacuje złożoność czasową zastosowanych algorytmów wyszukiwania — wyjaśnia na przykładach różnice między różnymi sposobami przekazywania parametrów do funkcji
Jak ocenić złożoność obliczeniową algorytmu?	• Czasowa złożoność obliczeniowa • Pamięciowa złożoność obliczeniowa • Algorytm optymalny	— definiuje złożoność obliczeniową algorytmu	— szacuje złożoność czasową i pamięciową	— wyjaśnia, czym jest złożoność oczekiwana (średnia), optymistyczna i pesymistyczna	— rozróżnia pojęcia algorytmu naiwnego i optymalnego	— określa złożoność czasową i pamięciową algorytmów z zastosowaniem odpowiednich wzorów — ocenia efektywność algorytmów

Metody sortowania prostego	• Sortowanie bąbelkowe • Sortowanie przez wybieranie • Sortowanie przez wstawanie	— definiuje pojęcie sortowania, prawidłowo określając klucz i porządek sortowania	— definiuje pojęcia sortowania *in situ* i stabilnego — stosuje metody sortowania prostego do sortowania liczb w zbiorze – bąbelkowe i przez wybieranie	— szacuje złożoność obliczeniową stosowanych algorytmów — definiuje operacje kluczowe (dominujące) w algorytmach sortowania	— pisze programy realizujące poznane algorytmy sortowania — podaje przykłady sortowania prostego w życiu codziennym — dobiera właściwe struktury danych	— definiuje własne funkcje do rozwiązywania problemów z wykorzystaniem algorytmów sortowania — ocenia wpływ pierwotnego ułożenia danych w zbiorze na liczbę wykonywanych operacji
Szyfry przestawieniowe, anagramy	• Anagramy • Szyfry przestawieniowe	— omawia zasadę działania szyfrów przestawieniowych, wymienia przykłady takich szyfrów	— sprawdza, czy słowa (napisy) są anagramami — pisze funkcje sprawdzające	— wykorzystuje poznane wcześniej algorytmy sortowania i zliczania w rozwiązywaniu problemów	— pisze program wyszukujący anagramy w plikach tekstowych	— do rozwiązania problemu dobiera optymalny algorytm i struktury danych
Sito Eratostenesa	• Działanie sita Eratostenesa • Implementacja algorytmu sita	— opisuje algorytmy sprawdzające, czy liczba jest pierwsza	— omawia i stosuje algorytm sita Eratostenesa do wyszukiwania liczb pierwszych w	— Implementuje algorytm sita Eratostenesa	— określa złożoność obliczeniową algorytmu	— optymalizuje algorytm, dążąc do minimalnej złożoności obliczeniowej

			określonym przedziale liczbowym			
Szukamy różnych podciągów	• Wyszukiwanie długości najdłuższego spójnego podciągu niemalejącego • Wyszukiwanie najdłuższego spójnego podciągu niemalejącego	— definiuje pojęcia podciągu oraz podciągu spójnego	— znajduje w zbiorze podciągi o różnych własnościach	— oblicza długość najdłuższego niemalejącego spójnego podciągu oraz liczbę jego elementów	— wymienia i stosuje różne algorytmy znajdowania maksymalnej sumy elementów spójnych podciągów, oceniając ich złożoność obliczeniową — znajduje w zbiorze spójny podciąg o maksymalnej sumie i wypisuje jego elementy	— do rozwiązania problemu dobiera optymalny algorytm i struktury danych
W poszukiwaniu lidera i idola	• Algorytm wyszukiwania lidera • Algorytm wyszukiwania idola	— definiuje pojęcia idola w grupie i lidera w zbiorze — znajduje idola w grupie lub stwierdza jego brak	— określa, czy w zbiorze jest lider — omawia i implementuje w języku C++ algorytmy szukania idola oraz	— ocenia złożoność obliczeniową stosowanych algorytmów i ich efektywność	— stosuje funkcję sort z biblioteki STL do wyszukiwania lidera	— do rozwiązania problemu dobiera optymalny algorytm i struktury danych

			lidera	— stosuje tablice dwuwymiarowe w pisanych programach		
Mini matura	Obejmuje wszystkie wymagania ujęte w wymaganiach edukacyjnych niezbędnych do otrzymania przez ucznia śródrocznej i rocznej oceny klasyfikacyjnej	— suma uzyskanych przez ucznia punktów mieści się w przedziale: 40% - 50%	— suma uzyskanych przez ucznia punktów mieści się w przedziale: 51% - 71%	— suma uzyskanych przez ucznia punktów mieści się w przedziale: 72% - 89%	— suma uzyskanych przez ucznia punktów mieści się w przedziale: 90% - 98%	— suma uzyskanych przez ucznia punktów mieści się w przedziale: 99% - 100%

UWAGI:

1. Ocenę wyższą otrzymuje uczeń spełniający łącznie wymagania edukacyjne określone dla ocen niższych np. ocenę dobrą otrzymuje uczeń spełniający wymagania edukacyjne na oceną dopuszczającą, dostateczną oraz dobrą.
2. Ocenę niedostateczną otrzymuje uczeń, który nie spełnia wymagań na poszczególne pozytywne oceny.
3. W przypadku nie zrealizowania tematów lekcji (zagadnień) w I okresie będą one realizowane po klasyfikacji śródrocznej. W tym przypadku obowiązują również wymagania edukacyjne dla tych tematów (zagadnień).