

Wymagania edukacyjne dla klasy 8 szkoły podstawowej zgodne z podręcznikiem „Lubię to!” (C++)

Wymagania zamieszczone poniżej zostały dostosowane do poszczególnych jednostek lekcyjnych i mają na celu ułatwienie planowania lekcji i oceniania uczniów. Są one propozycją, którą każdy nauczyciel powinien zmodyfikować stosownie do możliwości swojego zespołu klasowego.

Tytuł w podręczniku	Numer i temat lekcji	ocena dopuszczająca Uczeń:	ocena dostateczna Uczeń:	ocena dobra Uczeń:	ocena bardzo dobra Uczeń:	ocena celująca Uczeń:
DZIAŁ 1. Arkusz kalkulacyjny						
1.1. Formuły i adresowanie względne w arkuszu kalkulacyjnym	1. i 2. Formuły i adresowanie względne w arkuszu kalkulacyjnym	• omawia zastosowanie oraz budowę arkusza kalkulacyjnego • określa adres komórki • wprowadza dane różnego rodzaju do komórek arkusza kalkulacyjnego • formatuje zawartość komórek (wyrównanie tekstu oraz wygląd czcionki)	• określa zasady wprowadzania danych do komórek arkusza kalkulacyjnego • dodaje i usuwa wiersze oraz kolumny w tabeli	• tworzy proste formuły obliczeniowe • wyjaśnia, czym jest adres względny	• kopiuje utworzone formuły obliczeniowe, wykorzystując adresowanie względne	• samodzielnie tworzy i kopiuje skomplikowane formuły obliczeniowe
1.2. Funkcje oraz adresowanie bezwzględne i mieszane w arkuszu kalkulacyjnym	3. i 4. Funkcje oraz adresowanie bezwzględne i mieszane w arkuszu kalkulacyjnym	• rozumie różnice między adresowaniem względnym, bezwzględnym i mieszanym	• stosuje w arkuszu podstawowe funkcje: (SUMA, ŚREDNIA), wpisuje je ręcznie oraz korzysta z kreatora	• wykorzystuje funkcję JEŻELI do tworzenia algorytmów z warunkami w arkuszu kalkulacyjnym • ustawia format danych komórki odpowiadający jej zawartości • w formułach stosuje adresowanie względne, bezwzględne i mieszane	• korzysta z biblioteki funkcji, aby wyszukiwać potrzebne funkcje • stosuje adresowanie względne, bezwzględne lub mieszane w zaawansowanych formułach obliczeniowych	• stosuje zaawansowane funkcje arkusza w tabelach stworzonych na własne potrzeby
1.3. Przedstawianie danych na wykresie	5. i 6. Przedstawianie danych na wykresie	• wstawia wykres do arkusza kalkulacyjnego	• omawia i modyfikuje poszczególne elementy wykresu	• dobiera odpowiedni wykres do rodzaju danych	• tworzy wykres dla więcej niż jednej serii danych	• tworzy rozbudowane wykresy dla wielu serii danych

1.4. Zastosowania arkusza kalkulacyjnego	7. 8. Zastosowania arkusza kalkulacyjnego	• korzysta z arkusza kalkulacyjnego w celu stworzenia kalkulacji wydatków	• zapisuje w tabeli arkusza kalkulacyjnego dane otrzymane z prostych doświadczeń i przedstawia je na wykresie	• sortuje oraz filtruje dane w arkuszu kalkulacyjnym	• tworzy prosty model (na przykładzie rzutu sześcienną kostką do gry) w arkuszu kalkulacyjnym • stosuje filtry niestandardowe	• przygotowuje rozbudowane arkusze kalkulacyjne korzysta z arkusza kalkulacyjnego do analizowania doświadczeń z innych przedmiotów
DZIAŁ 3. Programowanie w języku C++						
3.1. Wprowadzenie do programowania w języku C++	9., 10. i 11. Wprowadzenie do programowania w języku C++	• definiuje pojęcia: algorytm, program, programowanie • podaje kilka sposobów przedstawienia algorytmu	• wymienia różne sposoby przedstawienia algorytmu: opis słowny, lista kroków • poprawnie formułuje problem do rozwiązania • stosuje odpowiednie polecenie języka C++, aby wyświetlić tekst na ekranie • omawia różnice pomiędzy kodem źródłowym a kodem wynikowym • tłumaczy, czym jest środowisko programistyczne	• wymienia przykładowe środowiska programistyczne • wyjaśnia, czym jest specyfikacja problemu • opisuje etapy rozwiązywania problemów • opisuje etapy powstawania programu komputerowego • zapisuje proste polecenia języka C++	• pisze proste programy w języku C++	• zapisuje algorytmy różnymi sposobami oraz pisze programy o większym stopniu trudności
3.2. Piszemy programy w języku C++	12., 13. i 14. Piszemy programy w języku C++	• tłumaczy, do czego używa się zmiennych w programach • pisze proste programy w języku C++ z wykorzystaniem zmiennych	• wykonuje obliczenia w języku C++ • omawia działanie operatorów arytmetycznych • stosuje tablice w języku C++ oraz operatory logiczne	• wykorzystuje instrukcję warunkową <code>if</code> oraz <code>if else</code> w programach • wykorzystuje iterację w konstruowanych algorytmach • wykorzystuje w programach instrukcję iteracyjną <code>for</code> • definiuje funkcje w języku C++ i omawia różnice między funkcjami zwracającymi wartość a funkcjami	• buduje złożone schematy blokowe służące do przedstawiania skomplikowanych algorytmów • konstruuje złożone sytuacje warunkowe (wiele warunków) w algorytmach • pisze programy zawierające instrukcje warunkowe, pętle oraz funkcje • wyjaśnia, jakie błędy zwraca interpreter	• pisze programy w języku C++ do rozwiązywanie zadań matematycznych • tworzy program składający się z kilku funkcji wywoływanych w programie głównym

				niezwracającymi wartości	• czyta kod źródłowy i opisuje jego działanie	
3.3. Algorytmy na liczbach naturalnych	15., 16. i 17. Algorytmy na liczbach naturalnych	• wyjaśnia działanie operatora modulo • wyjaśnia algorytm badania podzielności liczb	• zapisuje w postaci listy kroków algorytm badania podzielności liczb naturalnych • wykorzystuje w programach instrukcję iteracyjną <code>while</code>	• omawia algorytm Euklidesa i zapisuje go w wybranej postaci • wyjaśnia algorytm wyodrębniania cyfr danej liczby i zapisuje go w wybranej postaci	• wyjaśnia różnice między instrukcją iteracyjną <code>while</code> a pętlą <code>for</code> • pisze programy obliczające NWD, stosując algorytm Euklidesa, oraz wypisujące cyfry danej liczby	• pisze programy wykorzystujące algorytmy Euklidesa (np. obliczający NWW) oraz wyodrębniania cyfr danej liczby
3.4. Algorytmy wyszukiwania	18. i 19. Algorytmy wyszukiwania	• wyjaśnia potrzebę wyszukiwania informacji w zbiorze • sprawdza działanie programów wyszukujących element w zbiorze	• zapisuje algorytm wyszukiwania elementu w zbiorze nieuporządkowanym • implementuje grę w zgadywanie liczby	• implementuje algorytm wyszukiwania elementu w zbiorze nieuporządkowanym	• samodzielnie zapisuje w wybranej postaci algorytm wyszukiwania elementu w zbiorze	• samodzielnie modyfikuje i optymalizuje algorytmy wyszukiwania
3.5. Algorytmy porządkowania	20. i 21. Algorytmy porządkowania	• wyjaśnia potrzebę porządkowania danych • sprawdza działanie programu sortującego dla różnych danych	• zapisuje w wybranej formie algorytm porządkowania metodą przez wybieranie • omawia implementację algorytmu sortowania przez wybieranie • stosuje pętle zagnieżdżone i wyjaśnia, jak działają	• omawia funkcje zastosowane w kodzie źródłowym algorytmu sortowania przez wybieranie	• implementuje algorytm porządkowania metodą przez wybieranie • wprowadza modyfikacje w implementacji algorytmu porządkowania przez wybieranie	• samodzielnie modyfikuje i optymalizuje program sortujący metodą przez wybieranie
• DZIAŁ 4. Projekty						
4.1. Dokumentacja szkolnej imprezy sportowej	22. i 23. Dokumentacja szkolnej imprezy sportowej	• bierze udział w przygotowaniu dokumentacji szkolnej imprezy sportowej, wykonując powierzone mu zadania o niewielkim stopniu trudności	• bierze udział w przygotowaniu dokumentacji szkolnej imprezy sportowej • wprowadza dane do zaprojektowanych tabel	• przygotowuje dokumentację imprezy, wykonuje obliczenia, projektuje tabele oraz wykresy • współpracuje w grupie podczas pracy nad projektem	• bierze udział w przygotowaniu dokumentacji szkolnej imprezy sportowej, przygotowuje zestawienia, drukuje wyniki	• bierze udział w przygotowaniu dokumentacji szkolnej imprezy sportowej, tworzy zestawienia zawierające zaawansowane formuły, wykresy oraz elementy graficzne

					• współpracuje w grupie podczas pracy nad projektem	• współpracuje w grupie podczas pracy nad projektem, przyjmuje funkcję lidera
4.2. Sterowanie obiektem na ekranie	24., 25. i 26. Sterowanie obiektem na ekranie	• aktywnie uczestniczy w pracach zespołu, realizuje powierzone zadania o niewielkim stopniu trudności • testuje grę na różnych etapach • współpracuje w grupie podczas pracy nad projektem	• bierze udział w pracach nad wypracowaniem koncepcji gry • współpracuje w grupie podczas pracy nad projektem	• programuje wybrane funkcje i elementy gry • opracowuje opis gry	• implementuje i optymalizuje kod źródłowy gry, korzystając z wypracowanych założeń	• rozbudowuje grę o nowe elementy • współpracuje w grupie podczas pracy nad projektem, przyjmuje funkcję lidera