Wymagania edukacyjne Informatyka – semestr III (Liceum wieczorowe / stacjonarne)

Temat	Wymagania konieczne (ocena dopuszczająca)	Wymagania podstawowe (ocena dostateczna)	Wymagania rozszerzające (ocena dobra)	Wymagania dopełniające (ocena bardzo dobra)	Wymagania wykraczające (ocena celująca)
	Słuchacz:	**Słuchacz:**	**Słuchacz:**	**Słuchacz:**	**Słuchacz:**
1. Lekcja organizacyjna BHP pracy z komputerem, zasady promowania i oceniania	Zapoznanie z Przedmiotowym Systemem Oceniania (PSO) w semestrze III (sieci komputerowe, zaawansowany arkusz kalkulacyjny, grafika wektorowa, funkcje i tablice/listy w programowaniu), zasadami BHP, kryteriami sprawdzianu dopuszczeniowego i egzaminu semestralnego.				
Moduł D Wokół Internetu i projektów					
2. Temat D1. Internet jako „ocean informacji" oraz Temat D2. Korzystanie z wybranych e-usług	– wyszukuje proste informacje w wyszukiwarce internetowej – podaje przykłady e-usług (e-recepta, e-bankowość) – bezpiecznie loguje się do szkolnej poczty elektronicznej	– stosuje operatory logiczne w wyszukiwarkach (cudzysłów, minus, site:) – krytycznie weryfikuje wiarygodność informacji i źródeł w Internecie (odróżnia fake news) – charakteryzuje i bezpiecznie korzysta z e-usług publicznych (ePUAP, mObywatel, profil zaufany) – zna zasady bezpiecznego korzystania z bankowości internetowej	– sprawnie odnajduje specjalistyczne informacje branżowe i naukowe z użyciem zaawansowanych filtrów – identyfikuje metody manipulacji informacją i dezinformacji w mediach społecznościowych – posługuje się podpisem zaufanym i certyfikatem kwalifikowanym do załatwiania spraw urzędowych online – charakteryzuje mechanizmy zabezpieczeń płatności internetowych (BLIK, 3D Secure)	– przeprowadza pogłębioną analizę zjawiska bańki informacyjnej i mechanizmów profilowania użytkowników – bezpiecznie zarządza tożsamością cyfrową w relacjach z administracją i instytucjami finansowymi – potrafi skonfigurować profil zaufany i zarządzać uprawnieniami aplikacji e-usług	– analizuje wpływ algorytmów rekomendacyjnych i sztucznej inteligencji na dostęp do informacji w sieci – projektuje procedury bezpiecznego korzystania z e-usług i ochrony prywatności w przedsiębiorstwie – prowadzi warsztaty na temat przeciwdziałania dezinformacji i cyberzagrożeniom w e-usługach
Moduł A Wokół komputera					
3. Temat A1. Sieci komputerowe oraz Temat A2. Bezpieczeństwo i ochrona danych w komputerach i sieciach komputerowych	– podaje definicję sieci komputerowej – wymienia podstawowe urządzenia sieciowe (router, kabel sieciowy, Wi-Fi) – zna pojęcie wirusa komputerowego i programu antywirusowego – stosuje hasło do zabezpieczenia swojego profilu	– rozróżnia rodzaje sieci komputerowych (LAN, WAN, WLAN, Internet) – odczytuje adres IP komputera za pomocą polecenia `ipconfig` w konsoli – wymienia typowe zagrożenia sieciowe (malware, phishing, ransomware) – stosuje zasady tworzenia silnych	– charakteryzuje rolę routera, przełącznika (switcha) oraz karty sieciowej w sieci lokalnej – wyjaśnia strukturę adresu IPv4 oraz rolę maski podsieci i bramy domyślnej – konfiguruje zaporę sieciową (firewall) i skanowanie antywirusowe – wdraża uwierzytelnianie dwuskładnikowe (2FA) na kontach internetowych	– posługuje się narzędziami diagnostycznymi sieci w konsoli (`ping`, `tracert`, `netstat`) – analizuje wektory ataków socjotechnicznych i metody wyłudzania danych (phishing, spoofing) – konfiguruje bezpieczny punkt dostępowy Wi-Fi (standard WPA2/WPA3, ukrywanie SSID, filtracja MAC)	– projektuje logiczną i fizyczną topologię sieci komputerowej dla małego biura – charakteryzuje protokoły trasowania, model TCP/IP oraz rolę serwerów DNS i DHCP – opracowuje kompleksową politykę bezpieczeństwa informacji i ochrony przed cyberatakami w organizacji

		haseł i aktualizuje oprogramowanie antywirusowe			
Moduł B Wokół dokumentów komputerowych					
4. Temat B1. Formuły, funkcje i wykresy w arkuszu kalkulacyjnym	– wprowadza dane liczbowe i tekstowe do komórek arkusza kalkulacyjnego – stosuje podstawowe formuły arytmetyczne (+, -, *, /) – wstawia funkcję SUMA za pomocą przycisku Autosumowanie – tworzy prosty wykres kolumnowy	– bezbłędnie stosuje adresowanie względne i bezwzględne komórek (znak `$`, np. `$A$1`) – posługuje się funkcjami statystycznymi: ŚREDNIA, MAX, MIN, ILE.LICZB – stosuje funkcję logiczną JEŻELI do podejmowania decyzji dwuwariantowej – dobiera odpowiedni typ wykresu (kolumnowy, kołowy, liniowy) i dodaje tytuł oraz legendę	– łączy funkcje w wyrażeniach złożonych (np. funkcja JEŻELI zagnieżdżona lub z funkcjami ORAZ/LUB) – stosuje formatowanie warunkowe do wyróżniania komórek spełniających określone kryteria – precyzyjnie konfiguruje serie danych, osie wykresu, etykiety danych i linie trendu – formatuje arkusz do druku (nagłówki stron, obszar wydruku)	– projektuje profesjonalne arkusze kalkulacyjne z dynamicznym przeliczaniem danych i symulacją scenariuszy – bezbłędnie stosuje funkcje matematyczne i zaokrągleń (ZAOKR, SUMA.JEŻELI, LICZ.JEŻELI) – zabezpiecza komórki zawierające formuły przed przypadkową modyfikacją (blokowanie i ochrona arkusza)	– tworzy zaawansowane modele finansowo-ekonomiczne (kalkulatory rat, harmonogramy spłat, symulacje inwestycyjne) – łączy funkcje wyszukiwania i odwołań (WYSZUKAJ.PIONOWO, INDEKS, PODAJ.POZYCJĘ) – optymalizuje strukturę arkusza pod kątem automatyzacji i bezbłędności wprowadzania danych
5. Temat B2. Filtry oraz tabele i tabele przestawne w arkuszu kalkulacyjnym	– porządkuje dane alfabetycznie lub rosnąco za pomocą przycisku Sortuj – włącza autofiltr w nagłówkach kolumn – wybiera z listy prosty filtr tekstowy	– wykonuje sortowanie wielopoziomowe według kilku kolumn – stosuje filtry liczb (większe niż, pomiędzy, 10 pierwszych) i dat – formatuje zakres danych jako oficjalną tabelę Excela i włącza wiersz sumy – tworzy prostą tabelę przestawną z podsumowaniem jednej kolumny	– konfiguruje pola tabeli przestawnej (wiersze, kolumny, wartości, filtry raportu) – zmienia funkcję podsumowującą w tabeli przestawnej (suma, średnia, licznik, procent z sumy końcowej) – grupuje dane w tabeli przestawnej (np. daty według miesięcy i kwartałów) – tworzy i formatuje wykres przestawny powiązany dynamicznie z tabelą	– wykonuje zaawansowaną analizę wielowymiarową dużych zbiorów danych za pomocą tabel przestawnych – wstawia fragmentatory (slicers) i osie czasu do interaktywnego filtrowania raportu – tworzy pola i elementy obliczeniowe wewnątrz tabeli przestawnej	– projektuje profesjonalny, interaktywny pulpit menedżerski (dashboard) oparty na tabelach i wykresach przestawnych – łączy relacyjnie dane z wielu tabel arkusza w jeden model danych raportu przestawnego – demonstruje pełną biegłość analityczną w wyciąganiu wniosków biznesowych ze złożonych baz danych
6. Sprawdzian dopuszczeniowy	Praktyczny sprawdzian przy stanowisku komputerowym weryfikujący umiejętności z zakresu arkusza kalkulacyjnego (formuły, adresowanie bezwzględne, funkcje logiczne, tabele przestawne) oraz sieci komputerowych – warunek konieczny dopuszczenia do egzaminu semestralnego.				
7. Temat B3. Opracowywanie grafiki wektorowej	– uruchamia program grafiki wektorowej (np. Inkscape) – rysuje podstawowe kształty geometryczne (prostokąt, koło) – zmienia kolor wypełnienia i konturu obiektu	– wyjaśnia różnicę między grafiką rastrową a wektorową (skalowalność bez utraty jakości) – rozpoznaje i stosuje formaty plików wektorowych (SVG, PDF, EPS) – przekształca figury w ścieżki i modyfikuje położenie węzłów	– wykonuje operacje logiczne na obiektach wektorowych: sumę, różnicę, część wspólną, wykluczenie – precyzyjnie operuje uchwytami węzłów krzywych Béziera, tworząc gładkie kontury – stosuje wyrównywanie, rozkładanie obiektów w przestrzeni	– projektuje profesjonalny znak graficzny, ikonę lub ilustrację wektorową o spójnej kompozycji – stosuje wypełnienia gradientowe (liniowe, promieniowe) z kontrolą przezroczystości – wykonuje wektoryzację	– tworzy zaawansowaną kompozycję wektorową (infografikę, plakat reklamowy, postać) o wysokiej estetyce – bezbłędnie przygotowuje pliki wektorowe do profesjonalnego cięcia ploterowego i druku wielkoformatowego

		– rysuje proste krzywe za pomocą narzędzia Pióro (krzywe Béziera)	roboczej oraz grupowanie – projektuje proste logo wektorowe i eksportuje je do formatu SVG i rastrowego PNG	(trasowanie) mapy bitowej do postaci wektorowej	– wykazuje mistrzowską precyzję w operowaniu węzłami i ścieżkami wektorowymi
Moduł C Wokół algorytmiki i programowania					
8. Temat C1. Więcej o stosowaniu instrukcji iteracyjnych w językach C++ i Python	– rozpoznaje pętlę for i while w kodzie programu – uruchamia program z pętlą wykonującą określoną liczbę obiegów – wskazuje instrukcję odpowiedzialną za powtarzanie	– zna działanie i stosuje instrukcję `break` do wcześniejszego przerwania pętli – zna działanie i stosuje instrukcję `continue` do pominięcia bieżącej iteracji – wyjaśnia zasadę działania pętli zagnieżdżonych (pętla zewnętrzna i wewnętrzna) – pisze program generujący prosty ciąg liczb za pomocą pętli	– implementuje pętle zagnieżdżone do generowania dwuwymiarowych zestawień (np. tabliczka mnożenia) – tworzy algorytmy rysujące figury geometryczne ze znaków (kwadrat, trójkąt, ramka) – poprawnie kontroluje warunki zakończenia pętli wielokrotnych bez ryzyka zacięcia programu	– projektuje zaawansowane algorytmy iteracyjne przeglądające przestrzenie dwuwymiarowe – stosuje klauzulę `else` do pętli w Pythonie (kod wykonywany po normalnym zakończeniu pętli) – optymalizuje liczbę operacji w pętlach zagnieżdżonych	– tworzy algorytmy generujące złożone fraktale znakowe lub macierze transformacji – formalnie dowodzi poprawności i skończoności skomplikowanych struktur iteracyjnych – wykazuje biegłość w konstruowaniu wydajnych pętli obliczeniowych w Pythonie i C++
9. Temat C2. Stosowanie funkcji w językach C++ i Python	– rozpoznaje nagłówek funkcji w kodzie programu (`def` w Pythonie) – wywołuje gotową funkcję bezargumentową – modyfikuje treść wewnątrz ciała funkcji	– wyjaśnia pojęcie dekompozycji programu na funkcje pomocnicze – definiuje i wywołuje własną funkcję z parametrami formalnymi – stosuje instrukcję `return` do przekazania wyniku obliczeń z funkcji – odróżnia zmienne lokalne (wewnątrz funkcji) od zmiennych globalnych	– projektuje funkcje realizujące konkretne zadania obliczeniowe i logiczne (np. sprawdzanie podzielności, obliczanie pola) – stosuje parametry domyślne w funkcjach języka Python – zwraca z funkcji wiele wartości (w krotce w Pythonie) – unika niepożądanych efektów ubocznych modyfikowania zmiennych globalnych	– tworzy spójną bibliotekę modularnych funkcji wielokrotnego użytku – dokumentuje funkcje za pomocą profesjonalnych napisów dokumentacyjnych (docstring) – przekazuje funkcje jako argumenty do innych funkcji (podstawy programowania funkcyjnego)	– projektuje architekturę programu z pełną dekompozycją funkcyjną i enkapsulacją logiki biznesowej – posługuje się funkcjami anonimowymi (`lambda`) oraz dekoratorami w języku Python – analizuje działanie stosu wywołań funkcji i optymalizację przekazywania argumentów przez wartość i referencję
10. Temat C3. Stosowanie tablic w języku C++ i list w języku Python	– tworzy prostą listę w Pythonie z kilkoma elementami liczbowymi – odczytuje pierwszy element listy za pomocą indeksu `[0]` – wyświetla całą listę na ekranie	– wyjaśnia pojęcie struktury danych: tablica jednowymiarowa (C++) oraz lista (Python) – odczytuje i modyfikuje elementy listy/tablicy przez ich indeksy – dodaje nowe elementy do listy za pomocą metody `append()` i sprawdza długość listy funkcją `len()` – przegląda elementy listy za pomocą pętli `for`	– usuwa elementy z listy za pomocą metod `remove()`, `pop()` lub słowa kluczowego `del` – swobodnie posługuje się wycinaniem fragmentów list (slicing: `[start:stop:step]`) – tworzy program wczytujący n liczb od użytkownika i zapisujący je w strukturze danych – sprawdza przynależność elementu do listy operatorem `in`	– tworzy i przetwarza listy za pomocą zaawansowanych list składanych (list comprehension) – sprawnie kopiuje listy (odróżnia kopię płytką od głębokiej) – łączy i scala wiele list oraz stosuje metody sortowania wbudowanego (`sort()`, `sorted()`)	– projektuje i implementuje zaawansowane struktury danych (stos, kolejka FIFO) oparte na listach – analizuje złożoność pamięciową i czasową operacji na tablicach dynamicznych w Pythonie i statycznych w C++ – demonstruje pełną biegłość w optymalnym zarządzaniu

					kolekcjami danych w programowaniu
11. Temat C3. Przetwarzanie danych w strukturach tablicowych	– uruchamia program obliczający sumę elementów listy – wprowadza własne liczby do zdefiniowanej listy w kodzie – odczytuje największą liczbę wypisaną przez program	– implementuje algorytm sumowania elementów tablicy/listy w pętli – oblicza średnią arytmetyczną liczb zapisanych w liście – programuje algorytm wyszukiwania wartości największej (maksimum) i najmniejszej (minimum) w liście	– implementuje algorytm wyszukiwania liniowego zadanego elementu w liście (zwracanie indeksu lub informacji o braku) – programuje zliczanie elementów spełniających określony warunek (np. ile liczb dodatnich, ile parzystych) – tworzy nową listę zawierającą wyłącznie elementy odfiltrowane z listy źródłowej	– implementuje algorytm wyszukiwania jednocześnie minimum i maksimum z minimalizacją liczby porównań – programuje algorytm odwracania kolejności elementów w tablicy bez używania dodatkowej tablicy pomocniczej – zabezpiecza program przed błędami operacji na pustej liście (`IndexError`)	– projektuje i programuje wydajne algorytmy przetwarzania wielowymiarowych struktur tablicowych (macierzy) – optymalizuje algorytmy wyszukiwania pod kątem minimalizacji liczby odwołań do pamięci – przeprowadza rygorystyczne testy wydajnościowe przetwarzania dużych wolumenów danych
12. Temat C4. Wybrane algorytmy na liczbach naturalnych i algorytmy porządkowania	– podaje definicję dzielnika liczby i liczby pierwszej – oblicza ręcznie NWD dwóch małych liczb (np. 12 i 18) – uruchamia program wyznaczający NWD	– omawia zasadę działania algorytmu Euklidesa w wersji z odejmowaniem – implementuje algorytm Euklidesa w postaci iteracyjnej w języku Python lub C++ – implementuje sprawdzanie, czy podana liczba naturalna jest liczbą pierwszą (test naiwny w pętli)	– implementuje zoptymalizowaną wersję algorytmu Euklidesa z wykorzystaniem operacji reszty z dzielenia (modulo `%`) – programuje wyznaczanie Najmniejszej Wspólnej Wielokrotności (NWW) z wykorzystaniem zależności $NWW(a, b) = (a \cdot b) / NWD(a, b)$ – optymalizuje algorytm sprawdzania liczb pierwszych (badanie dzielników tylko do pierwiastka z liczby $\sqrt{n}$)	– implementuje algorytm sita Eratostenesa do wyznaczania liczb pierwszych w zadanym przedziale $[2, n]$ – analizuje liczbę kroków i porównuje efektywność obu wersji algorytmu Euklidesa – programuje wyznaczanie wszystkich dzielników podanej liczby naturalnej	– formalnie dowodzi poprawności matematycznej algorytmu Euklidesa – implementuje zaawansowane testy pierwszości i generatory liczb pierwszych wykorzystywane w kryptografii – wykazuje biegłość w algorytmicznej teorii liczb i optymalizacji kodu obliczeniowego
13. Temat C4. Implementacja algorytmu sortowania	– wyjaśnia, na czym polega uporządkowanie danych (rosnąco/malejąco) – podaje ręcznie posortowany ciąg kilku liczb – uruchamia gotowy program sortujący liczby	– wyjaśnia ideę działania algorytmu sortowania bąbelkowego (Bubble Sort) – omawia rolę pętli zagnieżdżonych w algorytmie sortowania – implementuje prosty algorytm sortowania bąbelkowego dla tablicy/listy liczb w Pythonie lub C++	– optymalizuje algorytm sortowania bąbelkowego (zmniejszanie zakresu pętli wewnętrznej o elementy już ustawione) – wprowadza zmienną logiczną (flagę) przerywającą sortowanie, gdy zbiór jest już posortowany – zlicza liczbę wykonanych porównań i przestawień elementów podczas sortowania	– implementuje alternatywny klasyczny algorytm sortowania: sortowanie przez wybieranie (Selection Sort) lub wstawianie (Insertion Sort) – porównuje efektywność zaimplementowanych algorytmów sortowania dla danych losowych, posortowanych i pesymistycznych – programuje sortowanie obiektów według wybranego kryterium (np. sortowanie uczniów według ocen)	– charakteryzuje i implementuje szybkie algorytmy sortowania o złożoności $O(n \log n)$ (np. sortowanie przez scalanie lub QuickSort) – analizuje stabilność i pamięciochłonność algorytmów porządkowania danych – demonstruje głębokie zrozumienie teorii algorytmów i struktur danych

14. Egzamin semestralny	Praktyczny egzamin semestralny przy stanowisku komputerowym sprawdzający efekty kształcenia z semestru III: zadanie analityczne w arkuszu kalkulacyjnym (formuły, tabela przestawna) oraz zadanie programistyczne w Pythonie/C++ (struktury list/tablic, funkcje, klasyczne algorytmy).
15. Omówienie wyników egzaminu	Przedstawienie wyników egzaminu semestralnego, omówienie poprawnych rozwiązań, analiza błędów w formułach i kodzie, wystawienie ocen semestralnych i zatwierdzenie promocji na semestr IV.