Informatyka - klasa II i III

Wymagania edukacyjne

Wymagane umiejętności na poszczególne oceny szkolne				
Zasady działania komputera				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Określa następujące po-jęcia: bit, bajt. Zna pojęcie systemu pozycyjne-go. Wymienia części składowe zestawu komputerowego, podaje ich parametry i przeznaczenie. Rozróżnia rodzaje pamięci komputera, określa ich własno-ści i przeznaczenie. Wie, co to jest system operacyjny, i korzysta z jego podstawowych funkcji. Wykonuje podstawowe operacje na plikach i fol-derach.	Wie, co to jest system binarny, i potrafi dokonać zamiany liczby z systemu dziesiętnego na binarny i odwrotnie. Potrafi sklasyfikować środki (urządzenia) i narzędzia (oprogramowanie) technologii informacyjnej. Wie, jak działa komputer. Wyjaśnia rolę procesora. Rozumie organizację pamięci komputerowych. Potrafi omówić funkcje systemu operacyjnego. Zna zasady ochrony plików. Potrafi nadać podstawowe atrybuty plikom, jak też wyszukać poszczególne pliki.	Potrafi wykonać działania arytmetyczne na liczbach binarnych (dodawanie i odejmowanie). Zna system szes-nastkowy i potrafi wykonać konwersję liczb binarnych na liczby w systemie szesnastkowym i odwrotnie. Analizuje model logiczny komputera. Wie, co to jest kod ASCII. Potrafi wymienić rodzaje aktualnie używanych komputerów. Zna metody wyszukiwa-nia plików.	Potrafi omówić dokładnie działanie procesora. Potrafi wykonać dowolną konwersję pomiędzy systemem dziesiętnym, dwójkowym i szesnastkowym. Zna sposób zapisu liczby całkowitej i rzeczywistej (zmiennoprzecinkowej). Umie wymienić przy-najmniej dwa systemy operacyjne i podać ich najważniejsze funkcje. Zna zaawansowane metody wyszukiwania i odzyskiwania plików. Zna przynajmniej jeden algorytm szyfrowania danych. Potrafi zaszyfrować i odszyfrować prosty tekst.	Zna operacje logiczne na liczbach binarnych i przesunięcia bitowe. Potrafi zapisać w języku programowania wysokiego poziomu algorytm konwersji liczb z dowolnego systemu pozycyj-nego na inny. Wykonuje sprawnie operacje na liczbach zapisanych w różnych systemach pozycyjnych. Potrafi odzyskać utracony plik, stosując zaawansowane metody. Potrafi omówić różne systemy opera-cyjne, wskazać ich najważniejsze funkcje. Samodzielnie wyszukuje informacje na temat kompresji i szyfrowania danych. Zna kilka sposobów szyfrowania informacji. Potrafi zapisać algorytm szyfrowania w postaci programu. Zna działanie algorytmu kompresji.

Rozwiązywanie problemów algorytmicznych				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Wie, co to jest algorytm. Określa dane do zadania oraz wyniki. Zna podstawowe zasady graficznego prezentowania algorytmów: podstawowe rodzaje bloków, ich przeznaczenie i sposoby umieszczania w schemacie blokowym. Potrafi narysować (odręcznie) schemat blokowy algorytmu liniowego.	Wymienia przykłady czynności i działań w życiu codziennym oraz zadań szkolnych, które uważa się za algorytmy. Zna pojęcie specyfikacji zadania. Zna wybrane sposoby prezentacji algorytmów. Przedstawia algorytm w postaci listy kroków. Tworzy schemat blokowy algorytmu z warunkiem prostym i pętlą. Podczas rysowania schematów blokowych potrafi wykorzystać Autokształty z edytora tekstu. Korzysta z programu edukacyjnego do symulacji działania algorytmu skonstruowanego w postaci schematu blokowego.	Określa zależności między problemem, algorytmem a programem komputerowym. Potrafi odpowiedzieć na pytanie, czy istnieją działania, które nie mają cech algorytmów. Przedstawia dokładną specyfikację dowolnego zadania. Zna znaczenie i działanie instrukcji symbolicznego języka programowania (pseudojęzyka). Potrafi zapisać algorytm z warunkami zagnieżdżonymi i pętlą w wybranej postaci. Potrafi skonstruować algorytm z warunkami zagnieżdżonymi i pętlą za pomocą programu edukacyjnego.	Zapisuje dowolny algorytm w wybranej przez siebie postaci (notacji), m.in. w pseudojęzyku. Zapisuje algorytmy z pętlą zagnieżdżoną. Potrafi przeprowadzić szczegółową analizę poprawności konstrukcji schematu blokowego. Analizuje działanie algorytmu dla przykładowych danych. Stosuje swobodnie oprogramowanie edukacyjne do graficznej prezentacji i analizy algorytmów.	Przestrzega zasad zapisu algorytmów w zadanej postaci (notacji). Potrafi trafnie dobrać do algorytmu sposób prezentacji. Stosuje poznane metody prezentacji algorytmów w opisie zadań (problemów) z innych przedmiotów szkolnych oraz różnych dziedzin życia. Potrafi samodzielnie zapoznać się z nowym programem edukacyjnym przeznaczonym do konstrukcji schematów blokowych. Potrafi zaproponować własny pseudojęzyk.

Techniki algorytmiczne i algorytmy klasyczne				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Określa sytuacje warunkowe. Podaje przykłady zadań, w których występują sytuacje warunkowe. Wie, na czym polega powtarzanie tych samych operacji. Potrafi omówić, na przykładzie, algorytm znajdowania najmniejszego z trzech elementów.	Potrafi odróżnić algorytm liniowy od algorytmu z warunkami (z rozgałęzieniami). Zna pojęcie iteracji i rozumie pojęcie algorytmu iteracyjnego. Podaje ich przykłady. Wie, od czego zależy liczba powtórzeń. Potrafi omówić algorytm porządkowania elementów (metodą przez wybór) na praktycznym przykładzie, np. wybierając najwyższego ucznia z grupy. Omawia i analizuje wybrane techniki sortowania w postaci gotowych schematów blokowych, skonstruowanych w programie edukacyjnym.	Analizuje algorytmy, w których występują iteracje. Zna sposoby zakończenia iteracji. Określa kroki iteracji. Potrafi zapisać w wybranej notacji np. algorytm sumowania n liczb, algorytm obliczania silni, znajdowania minimum w ciągu n liczb, algorytm rozwiązywania równania liniowego. Zna iteracyjną postać algorytmu Euklidesa. Zna przynajmniej dwie techniki sortowania, np. bąbelkowe i przez wybór. Określa problemy, w których występuje rekurencja i podaje przykłady „zjawisk rekurencyjnych” – wziętych z życia i zadań szkolnych. Zna rekurencyjną realizację wybranego algorytmu, np. silni, liczb Fibonacciego, NWD.	Zna metodę „dziel i zwyciężaj” , algorytm generowania liczb Fibonacciego, schemat Hornera. Omawia ich iteracyjną realizację i potrafi przedstawić jeden z nich w wybranej notacji. Zna inne algorytmy sortowania, np. kubełkowe, przez wstawianie. Zna przynajmniej jeden algorytm numeryczny, np. obliczanie wartości pierwiastka kwadratowego. Wskazuje różnicę między rekurencją a iteracją. Zna rekurencyjną realizację wybranych algorytmów, np. silnię i algorytm Euklidesa. Potrafi zamienić algorytm zapisany iteracyjnie na postać rekurencyjną.	Rozumie dokładnie technikę rekurencji (znaczenie stosu). Potrafi ocenić, kiedy warto stosować iterację, a kiedy rekurencję. Zna trudniejsze algorytmy, np. wieże Hanoi, problem ośmiu hetmanów. Zna inne techniki sortowania, np. sortowanie przez scalanie ciągów i metodę szybką. Potrafi zapisać je w różnych notacjach (również w języku programowania wysokiego poziomu). Zna inne algorytmy numeryczne, np. wyznaczanie miejsca zerowego funkcji. Korzysta samodzielnie z dodatkowej literatury.

Analiza algorytmów				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Potrafi zanalizować przebieg algorytmu dla przykładowych danych i ocenić w ten sposób jego poprawność.	Potrafi ocenić poprawność działania algorytmu i jego zgodność ze specyfikacją. Określa liczbę prostych działań zawartych w algorytmie.	Rozumie, co to jest złożoność algorytmu i potrafi określić liczbę operacji wykonywanych na elementach zbioru w wybranym algorytmie sortowania.	Potrafi porównać złożoność różnych algorytmów tego samego zadania dla tych samych danych. Wie, kiedy algorytm jest uniwersalny.	Ocenia złożoność czasową i pamięciową algorytmu. Zna odpowiednie wzory.
Zasady programowania w języku C++				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Zna klasyfikację języków programowania. Zna ogólną budowę programu w języku C++ i najważniejsze elementy języka – słowa kluczowe, instrukcje, wyrażenia, zasady składni. Potrafi zrealizować prosty algorytm liniowy w języku C++; potrafi skompilować i uruchomić gotowy program.	Zapisuje program w czytelnej postaci – stosuje wcięcia, komentarze. Rozumie pojęcia: implementacja, kompilacja, uruchomienie, testowanie. Rozumie znaczenie i działanie podstawowych instrukcji wybranego języka C++. Rozróżnia i poprawia błędy kompilacji i błędy wykonania. Potrafi zrealizować algorytmy iteracyjne w języku C++. Zna podstawowe zasady poprawnego programowania; testuje tworzone programy; wie, jak uniknąć problemów, takich jak np. zapętlenie się programu.	Potrafi prezentować złożone algorytmy (z podprogramami) w C++. Zna rekurencyjne realizacje prostych algorytmów. Rozumie i stosuje zasady programowania strukturalnego. Wie, na czym polega różnica pomiędzy przekazywaniem parametrów przez referencję i przez wartość w funkcjach C++. Wie, jakie znaczenie ma zasięg działania zmiennej. Rozumie zasady postępowania przy rozwiązywaniu problemu metodą zstępującą. Zna zasady działania wybranych algorytmów sortowania.	Wie, jaka jest różnica między językiem wysokiego poziomu a językiem maszynowym; potrafi określić rolę procesora i pamięci operacyjnej w działaniu programów. Potrafi realizować nawet bardzo złożone algorytmy. Potrafi prezentować algorytmy rekurencyjne w postaci programu; potrafi zamienić rozwiązanie iteracyjne algorytmu na rekurencyjne. Zapisuje w postaci programu wybrane algorytmy sortowania. Opracowuje złożony program w kilkuosobowej grupie – umie podzielić zadania, ustalić sposoby przekazywania danych pomiędzy procedurami.	Ocenia efektywność działania programu. Wie, na czym polega programowanie obiektowe i zdarzeniowe. Potrafi stosować techniki programowania dynamicznego lub programowania obiektowego. Zna i rozumie podobieństwa i różnice w strukturze programu zapisanego w różnych językach programowania – w deklaracji zmiennych i procedur, w składni i zasadach działania poszczególnych procedur. Sprawnie korzysta z dodatkowej, fachowej literatury.

Dobór struktur danych do rozwiązywanego problemu				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Wymienia przykłady prostych struktur danych. Potrafi zadeklarować zmienne typu liczbowego (całkowite, zmiennoprzecinkowe) i stosować je w zadaniach.	Wie, czym jest zmienna w programie i co oznacza przypisanie jej konkretnej wartości. Rozróżnia struktury danych: proste i złożone. Podaje przykłady. Deklaruje typy złożone.	Potrafi zastosować łańcuchowy i tablicowy typ danych w zadaniach.	Rozumie, na czym polega dobór struktur danych do algorytmu. Potrafi zastosować strukturalny typ danych.	Zna dynamiczne struktury danych. Potrafi zastosować zmienne typu wskaźnikowego w zadaniach. Zna struktury listowe, np. stos, kolejkę, listę, drzewo, graf. Potrafi je wykorzystać przy rozwiązywaniu określonych problemów, Rozumie i potrafi zastosować klasy obiektowe języka C++
Wymagane umiejętności na poszczególne oceny szkolne				
Sieci komputerowe i bazy danych				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Zna pojęcie sieci komputerowej, potrafi wymienić jej rodzaje. Zna pojęcie logowania. Potrafi wymienić kilka cech pracy w sieci, odróżniających ją od pracy na autonomicznym komputerze. Zna kilka sposobów połączenia z Internetem.	Wymienia korzyści płynące z korzystania z sieci. Zna podstawowe klasy i topologie sieciowe. Potrafi wymienić urządzenia i elementy sieciowe oraz omówić ich ogólne przeznaczenie. Zna cechy systemu działającego w szkolnej pracowni. Orientuje się – w zakresie podstawowym – w działaniu Internetu.	Zna znaczenie protokołu w sieciach lokalnych oraz TCP/IP. Wie, na czym polega wymiana informacji w sieci. Zna zasady pracy w sieci. Potrafi omówić zagrożenia płynące z sieci. Charakteryzuje różne połączenia z Internetem; potrafi omówić przesyłanie pakietów danych w Internecie.	Zna schemat działania sieci komputerowych. Potrafi wymienić zalety i wady różnych topologii sieci. Charakteryzuje topologie gwiazdy, magistrali i pierścienia. Umie z pomocą nauczyciela zrealizować małą sieć komputerową – skonfigurować jej składniki, udostępnić pliki, dyski, drukarki, dodać użytkowników.	Omawia szczegółowo model warstwowy sieci. Omawia różne systemy sieciowe. Dokonuje ich analizy porównawczej. Potrafi samodzielnie zbudować małą sieć domową.

Tendencje w rozwoju informatyki i jej zastosowań				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Potrafi omówić historię komputerów. Umie wskazać ogólny kierunek zmian w technologiach komputerowych. Zna podstawowe zasady netykiety.	Potrafi określić nowoczesne trendy w zastosowaniu urządzeń komputerowych. Jest w stanie omówić prawne i społeczne aspekty zastosowania informatyki.	Potrafi wskazać nowości w zakresie usług internetowych oraz odszukać informacje na temat najnowszych pomysłów na komputery.	Przygotowuje analizę porównawczą, pokazującą na przestrzeni wielu lat rozwój informatyki, w tym sieci komputerowych, oraz multimediów.	Wskazuje tendencje w rozwoju informatyki i jej zastosowania, dostrzegając przeobrażenia w tej dziedzinie w kraju i na świecie.
Relacyjne bazy danych				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Podaje obszary zastosowań baz danych – na przykładach z najbliższego otoczenia – szkoły, instytucji naukowych, społecznych i gospodarczych. Podaje przykłady programów do tworzenia baz danych. Potrafi wykonać podstawowe operacje na bazie danych przygotowanej w jednej tabeli (wprowadzanie, redagowanie, sortowanie, wyszukiwanie, prezentacja). Potrafi uporządkować bazę rosnąco lub malejąco według jednego lub kilku pól.	Rozumie metody przetwarzania danych na przykładzie gotowej bazy danych. Określa podstawowe pojęcia (rekord, pole, typ pola). Samodzielnie tworzy w jednej tabeli bazę danych, składającą się z kilku pól różnych typów. Projektuje przykładowy formularz i raport. Potrafi wykonywać operacje przetwarzania danych w bazie składającej się z kilku rekordów. Zna zasady przygotowania korespondencji seryjnej.	Projektuje relacyjną bazę danych składającą się z dwóch tabel połączonych relacją (na zadany temat). Projektuje formularz i raport według wskazówek nauczyciela. Zna zasady definiowania kluczy podstawowych. Drukuje wybrane rekordy, formularze i raporty. Łączy informacje z bazy danych z dokumentami innych programów, np. edytora tekstu czy arkusza kalkulacyjnego.	Potrafi wytłumaczyć pojęcie relacji. Projektuje relacyjną bazę danych składającą się z trzech lub większej liczby tabel. Samodzielnie ustala zawartość bazy (rodzaj informacji). Zna kilka rodzajów formularzy i raportów, w tym raporty w postaci wykresów. Umie zaprojektować samodzielnie wygląd formularza i raportu. Zna pojęcie indeksu. Odróżnia sortowanie od indeksowania. Potrafi w tworzonej bazie ustalić klucze indeksu.	Zna dokładnie wybrany program do projektowania baz danych. Potrafi samodzielnie zaprojektować bazę danych, korzystając z wybranego narzędzia (programu). Projekt bazy opiera na rzeczywistych informacjach, aby można było wykorzystać ją w praktyce, np. w szkole czy w domu. Sprawnie korzysta z dodatkowej, fachowej literatury.

Programowanie z wykorzystaniem SQL				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Wyszukuje informacje w bazie, korzystając wyłącznie z gotowych kwerend i narzędzi wbudowanych do programu.	Tworzy samodzielnie kwerendy (proste i złożone), korzystając z wbudowanych do programu narzędzi.	Zna podstawowe konstrukcje języka zapytań. Wie, co to jest język SQL. Potrafi przeanalizować przykład zapytania utworzonego w języku SQL. Z pomocą nauczyciela potrafi zapisać prostą kwerendę, korzystając z języka zapytań. Potrafi połączyć się z bazą danych z poziomu języka PHP.	Zna zasady wyszukiwania informacji w bazie z wykorzystaniem języka zapytań. Potrafi zapisać złożone kwerendy, korzystając z wybranej instrukcji, np. SELECT; stosuje jej główne klauzule. Z poziomu języka PHP potrafi pobierać dane z bazy i odpowiednio je prezentować na stronie WWW.	Opierając się na profesjonalnej literaturze, potrafi samodzielnie zapisywać złożone kwerendy z wykorzystaniem języka zapytań. Potrafi efektywnie wykorzystywać bazę danych z poziomu języka PHP – potrafi uaktualniać zapisane w bazie dane.
Tworzenie grafiki komputerowej				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Potrafi retuszować obrazy rastrowe, potrafi rysować proste figury wektorowe, potrafi dokonywać transformacji figur: skalowania, obrotów, ścinania, rozciągania, odbicia lustrzane, potrafi stosować różne wypełnienia figur wektorowych.	Potrafi tworzyć prostą grafikę wektorową, potrafi przemieszczać figury pomiędzy warstwami, potrafi edytować kształt figury, potrafi eksportować grafikę wektorową do pliku GIF lub JPG.	Rozumie różnicę pomiędzy plikami GIF i JPG, wie, na czym polega kompresja stratna w plikach JPG, potrafi tworzyć średnio złożoną grafikę z wykorzystaniem różnych narzędzi edytora.	Potrafi tworzyć skomplikowaną grafikę komputerową w edytorze graficznym, zna i stosuje różne funkcje zaawansowane programu, umie korzystać z przezroczystości oraz opcji powtarzania operacji.	Tworzy dowolną grafikę wektorową, wykorzystuje zaawansowane opcje edytora, w tym modele barw, filtry i transformacje graficzne.
Tworzenie animacji FLASH z wykorzystaniem ActionScript				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Potrafi wykorzystywać podstawowe narzędzia graficzne i tworzyć prostą animację poklatkową i pośrednią typu motion tween.	Potrafi posługiwać się transformacjami kształtu i wypełnienia, dodawać obrazy GIF i JPG, tworzyć animację na wielu warstwach.	Potrafi stosować w tworzonej grafice proste polecenia ActionScript, które wpływają na sposób odtwarzania filmu, umie wykorzystywać symbole biblioteczne.	Tworzy zaawansowane animacje w środowisku FLASH, stosuje skrypty ActionScript zawierające zmienne, potrafi wyeksportować animację do postaci pliku swf i użyć go na własnej stronie WWW.	Posługuje się obiektami w ActionScript, tworzy interaktywne animacje wykorzystujące zdarzenia generowane przez użytkownika.

Programowanie w JavaScript				
dopuszczający	dostateczny	dobry	bardzo dobry	celujący
Zna sposoby osadzania skryptów JavaScript w kodzie HTML strony WWW, potrafi tworzyć proste programy w JavaScript	Potrafi umieszczać programy w języku JavaScript w osobnych plikach i łączyć je z wieloma stronami WWW, umie umieszczać proste polecenia JavaScript w znacznikach HTML.	Potrafi stworzyć prostą aplikację obliczeniową w języku JavaScript, umie pobierać dane z formularzy i umieszczać wyniki obliczeń na stronach WWW.	Tworzy interaktywne strony WWW, które w pełni wykorzystują język JavaScript.	Tworzy zaawansowane aplikacje JavaScript, które wykorzystują obiekty i zdarzenia. Zna model obiektowy dokumentu HTML, umie posługiwać się własnościami obiektów HTML.