INFORMATYKA

KLASA I LO LICEUM OGÓLNOKSZTAŁCĄCE - plan wynikowy z wymaganiami edukacyjnymi przedmiotu

(podręcznik 1 część 2, zakres rozszerzony)

Temat (rozumiany jako lekcja)	Wymagania na ocenę **DOPUSZCZAJĄCĄ** Uczeń:	Wymagania na ocenę **DOSTATECZNĄ** Uczeń:	Wymagania na ocenę **DOBRĄ** Uczeń:	Wymagania dopełniające **BARDZO DOBRĄ** Uczeń:	Wymagania wykraczające **CELUJĄCĄ** Uczeń:
I. Podstawy programowania					
Edytor i kompilator, czyli środowisko zintegrowane (IDE) (1) Edytor programu Free Pascal (2)	– definiuje zintegrowane środowisko programistyczne; – korzystając z pliku pomocy i podręcznika, odnajduje najczęściej używane opcje dotyczące edycji programu; – wie, czym jest kompilacja;	– odnajduje i zna zastosowanie podstawowych opcji edytora Free Pascal; – edytuje tekst programu komputerowego w edytorze Free Pascal;	– sprawnie korzysta z możliwości edytora Turbo Pascal takich jak Search i Replace; – omawia istotę kodu źródłowego i jego kompilacji; – stosuje zasady zapisu programu w edytorze Free Pascal, stosując podział na linie i wcięcia w odpowiednich miejscach;	– samodzielnie i sprawnie korzysta ze wszystkich opcji edytora Free Pascal; – omawia funkcje wszystkich opcji środowiska programistycznego Free Pascal;	– zna i stosuje inne edytory dla programistów;
Tworzenie kodu źródłowego i budowa programu (3)	– zna strukturę programu w Pascalu z podziałem na bloki; – wie, czym są słowa kluczowe; – wie, na czym polega kompilacja programu;	– wskazuje w kodzie programu poszczególne części i opisuje ich funkcje; – wyjaśnia istotę blokowej struktury programu;	– wyjaśnia znaczenie i funkcje poszczególnych części programu komputerowego w Pascalu; – samodzielnie przeprowadza kompilację i uruchomienie programu komputerowego w języku Pascal; – korzysta z niektórych skrótów klawiszowych Free Pascal;	– kompiluje program, korzystając z pracy krokowej; – biegle korzysta ze skrótów klawiszowych środowiska Free Pascal;	– omawia różnice pomiędzy programowaniem strukturalnym a obiektowym; – w pełni korzysta z innych środowisk programistycznych podczas tworzenia kodu i kompilacji programu;

Identyfikatory w Turbo Pascalu (4)	– umie zdefiniować pojęcie identyfikatora; – zna jego znaczenie dla budowy programu; – definiuje nazwy zmiennych z pomocą nauczyciela;	– poprawnie definiuje nazwy identyfikatorów, używając właściwych znaków; – poprawnie używa identyfikatorów;	– poprawnie interpretuje komunikaty o błędach w definicjach nazw zmiennych, wykorzystując system pomocy i literaturę;	– właściwie dobiera nazwy zmiennych, biorąc pod uwagę ich przeznaczenie i przyjmowane wartości; – samodzielnie poprawnie interpretuje komunikaty o błędach związane z definiowaniem nazw zmiennych;	– bezbłędnie definiuje nazwy zmiennych w języku Pascal;
Typy proste (5) Zmienne i deklaracje (6) Stałe i definicje (7)	– zna różnice pomiędzy typami standardowymi a definiowanymi przez programistę; – wie, czym różnią się typy proste od złożonych; – zna podstawowe typy dla liczb całkowitych i rzeczywistych; – korzystając z pomocy nauczyciela, poprawnie dobiera podstawowe typy proste i definiuje je w programie; – zna najważniejsze cechy stałych i zmiennych;	– wymienia nazwy wszystkich typów prostych: całkowite, znakowy, logiczny i rzeczywiste; – definiuje typy, korzystając z tabeli z podręcznika; – zna przeznaczenie poszczególnych typów; – wie, czym różnią się stałe od zmiennych;	– samodzielnie definiuje najważniejsze typy; – rzadziej używane typy definiuje, korzystając z podręcznika; – wyświetla zawartość zmiennych i stałych;	– samodzielnie podaje zakresy liczb dla poszczególnych typów; – zna zasady definiowania wszystkich typów prostych w Pascalu; – Samodzielnie i trafnie dobiera nazwy i definiuje stałe;	– samodzielnie i biegle definiuje wszystkie typy proste w Pascalu;
Operatory i wyrażenia (8) Operatory i wyrażenia, typ znakowy i logiczny– ćwiczenia (9, 10)	– podaje definicję wyrażenia; – wie, jaka jest różnica pomiędzy wyrażeniem całkowitym a rzeczywistym; – zna podstawowe	– poprawnie i samodzielnie zapisuje proste wyrażenia na z wykorzystaniem operatorów; – poprawnie zapisuje i objaśnia operacje na	– korzystając z podręcznika lub odpowiednich tabel, zapisuje złożone wzory matematyczne za pomocą operatorów; – zna priorytety operatorów logicznych;	– samodzielnie tworzy dowolne wyrażenia łączące operatory stałe, zmienne i wartości funkcji; – sprawnie posługuje się operatorami div i mod;	– samodzielnie analizuje fragmenty programów zawierające wyrażenia, szacując ich wartość dla różnych wartości zmiennych i stałych; – szacuje wynik

	operatory; – zapisuje wyrażenia z liczbami całkowitymi za pomocą operatorów; – z pomocą nauczyciela poprawnie zapisuje wyrażenia z operatorami; – wymienia nazwy operacji na typach znakowych i z pomocą nauczyciela omawia ich przeznaczenie;	wartościach całkowitych, rzeczywistych, logicznych i znakowych;	– korzystając z opcji pomocy, analizuje komunikaty o błędach dotyczących wyrażeń;	– sprawnie stosuje operatory logiczne, pamiętając o priorytetach; – samodzielnie poprawnie odczytuje komunikaty o błędach dotyczących wyrażeń i poprawia błędy;	działania operatorów logicznych i znakowych; – samodzielnie i bezbłędnie zapisuje wszystkie rodzaje wyrażeń;
Instrukcje proste – zmieniamy wartość zmiennych (11)	– prawidłowo posługuje się instrukcją przypisania i zapisuje ją; – z pomocą nauczyciela układa proste programy z wprowadzaniem i wyprowadzaniem danych oraz prostymi obliczeniami;	– korzystając z podręcznika, przypisuje wartości jednej zmiennej do drugiej zmiennej; – układa proste programy bez kontroli zakresu zmiennych;	– samodzielnie i prawidłowo układa proste programy obliczające wartość wyrażenia z danymi wprowadzonymi do programu za pomocą klawiatury; – wyświetla wynik obliczeń na ekranie; – prawidłowo ocenia zakres wyniku obliczeń (z wykorzystaniem przykładów analizy z podręcznika);	– samodzielnie układa program z kontrolą zakresu zmiennych; – samodzielnie układa programy przypisujące wartości wyrażeń do zmiennych;	– celowo stosuje instrukcję pustą i uzasadnia swoją decyzję;
Instrukcje warunkowe – rozgałęziamy działania (12) Instrukcja wyboru – case (13)	– opisuje działanie instrukcji warunkowej; – z pomocą nauczyciela interpretuje algorytmy zawierające bloki decyzyjne;	– korzystając z przykładów w podręczniku, układa programy z instrukcją warunkową na podstawie algorytmu z blokiem decyzyjnym;	– samodzielnie układa programy z wykorzystaniem instrukcji warunkowej na podstawie algorytmu; – korzystając z opcji pomocy i przykładów z podręcznika, układa programy z instrukcjami	– samodzielnie układa programy z instrukcjami warunkowymi złożonymi i zagnieżdżonymi;	– samodzielnie opracowuje proste algorytmy z blokami decyzyjnymi i na ich podstawie układa programy w języku Pascal;

			warunkowymi złożonymi i zagnieżdżonymi;		
Instrukcje powtarzania – pętle for (14) Instrukcje powtarzania – pętle repeat … until (15) Instrukcje powtarzania – pętle while ...do (16)	– omawia i rozumie różnice pomiędzy for, repeat...until oraz while...do; – podaje przykłady zastosowań for, repeat...until oraz while...do; – definiuje zmienną sterującą i omawia jej znaczenie w for; – omawia znaczenie warunku w pętlach;	– umie wybrać rodzaj instrukcji powtarzania do rozwiązania określonego problemu; – pamięta o deklaracji zmiennej sterującej; – układa proste programy z pętlami, korzystając z przykładów rozwiązań podobnych problemów;	– prawidłowo układa fragmenty programów z zastosowaniem procedur i funkcji w pętlach instrukcji powtarzania, np. wczytywanie znaku z klawiatury, oczekiwanie na wciśnięcie klawisza itp.; – prawidłowo wybiera pomiędzy for, repeat...until oraz while...do instrukcję realizującą pętle z określoną lub nieokreśloną liczbą powtórzeń;	– układa programy, prawidłowo wykorzystując instrukcje powtarzania do budowy funkcjonalnego programu, np. w programie liczącym głosy; – używa instrukcji powtarzania do układania funkcji i procedur wywoływanych przez program główny, np. sprawdzania, czy został wciśnięty klawisz, podano prawidłowe hasło itp.;	– samodzielnie opracowuje algorytmy, które mogą być realizowane za pomocą instrukcji powtarzania, i na ich podstawie układa programy w języku Pascal; – przewiduje w algorytmach konieczność stosowania instrukcji for, repeat...until lub while...do;
Tablice – zaczynamy definiować swoje typy złożone (17) Tablice – definiujemy własne typy złożone (18) Tablice – definiujemy własne typy złożone – ćwiczenia (19)	– podaje definicję typu tablicowego i z pomocą nauczyciela podaje różne przykłady zastosowania tablic; – z pomocą nauczyciela analizuje definicję typu tablicowego, wskazując najważniejsze jej elementy; – obrazowo opisuje zastosowanie tablic jedno-, dwu-, trój- i więcej wymiarowych; – graficznie wyjaśnia budowę tablic;	– zna znaczenie słów kluczowych w definicjach tablic; – określa wartość zmiennej w tablicy za pomocą wartości typu indeksowego; – umie zaplanować tablicę o odpowiednim do rozwiązywanego problemu rozmiarze; – samodzielnie definiuje własne typy tablicowe jednowymiarowe;	– definiuje własne tablice zmiennych różnych typów i rozmiarów; – prawidłowo zastępuje zmienną tablicową osobno występujące zmienne, uzasadniając swoją decyzję i wykazując zasadność takiego postępowania; – podaje przykłady definiowania różnych tablic i objaśnia zasadność przyjęcia takiej konstrukcji zmiennej tablicowej, np. używanych w	– prawidłowo zastępuje zmienną tablicową osobno występujące zmienne, uzasadniając swoją decyzję i wykazując zasadność takiego postępowania; – nadaje wartości początkowe zmiennym tablicowym; – samodzielnie modyfikuje zmienne tablicowe, np. poprzez zwiększenie rozmiaru; – analizuje zajęcie pamięci przez daną tablicę; – umie wskazać błąd w	– samodzielnie opracowuje algorytmy, które wykorzystują tablice, i na ich podstawie układa programy w języku Pascal; – stosuje tablice wielowymiarowe do grupowania danych, np. wprowadzanych imion, nazwisk itp.;

			przykładzie z podręcznika;	deklaracjach tablic zarówno składniowy, jak i logiczny;	
Łańcuchy (20)	– zna zastosowanie zmiennych łańcuchowych i ich znaczenie; – podaje przykłady zastosowania zmiennych łańcuchowych; – analizuje poprawne definicje zmiennych łańcuchowych; – zna słowa kluczowe służące do deklaracji zmiennej łańcuchowej;	– prawidłowo deklaruje zmienne łańcuchowe; – poprawnie dobiera długość deklarowanej zmiennej typu string;	– dodaje zmienne łańcuchowe; – stosuje standardowe funkcje i procedury Pascala działające na stringach; – stosuje funkcje i procedury języka Pascal wykorzystujące zmienne łańcuchowe lenght, delete, pos, copy, concat, insert;	– uzasadnia wybór funkcji i procedur języka Pascal wykorzystujących zmienne łańcuchowe; – programuje wprowadzanie zmiennych łańcuchowych za pośrednictwem klawiatury; – szacuje zajętość pamięci na podstawie deklaracji zmiennej łańcuchowej;	– samodzielnie planuje użycie zmiennych łańcuchowych do rozwiązywania problemów informatycznych; – samodzielnie układa programy, stosując zmienne łańcuchowe w tablicach;
Procedury – piszemy własne podprogramy (21) Procedury – piszemy własne podprogramy – zmienne lokalne i globalne, przesłanianie zmiennych (22, 23) Funkcje – piszemy własne podprogramy (24)	– zna różnicę pomiędzy działaniem i zastosowaniem procedury i funkcji; – zna budowę procedury i funkcji; – wie, na czym polega wywołanie procedury i funkcji; – określa przypadki, w których niezbędne jest zastosowanie procedury lub funkcji;	– wywołuje procedury i funkcje w treści programu; – analizuje działanie procedury lub funkcji;	– stosuje unikalne nazwy zmiennych wewnątrz bloku; – prawidłowo układa mało skomplikowane procedury i funkcje wykorzystujące globalne i lokalne zmienne, np. obliczające; – wywołuje procedury z parametrami; – prawidłowo i czytelnie dobiera nazwy zmiennych w deklaracjach procedur i funkcji; – wie, jak działa przekazywanie przez wartość; – wie, jak działa	– omawia istotę deklaracji wewnętrznych zmiennych i stałych, typów i podprogramów lokalnych; – układa i wywołuje procedury z parametrami i bez; – prawidłowo układa procedury i funkcje wykorzystujące globalne i lokalne zmienne; – stosuje unikalne nazwy zmiennych wewnątrz bloku; – układa procedury z przekazywaniem parametrów przez wartość lub zmienną;	– samodzielnie planuje użycie funkcji i procedur do rozwiązywania problemów informatycznych; – samodzielnie układa procedury i funkcje, stosując zmienne lokalne i globalne;

			przekazywanie przez zmienną;		
Złożone struktury danych (rekordy) – grupujemy dane (25) Złożone struktury danych – zmienne rekordowe (26)	– określa funkcję jaką odgrywają rekordy w programach komputerowych; – odróżnia od siebie strukturę (typ rekordowy) i zmienną typu rekordowego; – z pomocą podręcznika analizuje przykłady typów rekordowych;	– definiuje rekord i typ rekordowy; – podaje przykłady rekordów i typów rekordowych; – definiuje zmienne rekordowe; – podaje przykłady zmiennych rekordowych;	– omawia strukturę zmiennych rekordowych; – deklaruje rekordy składające sie z różnych danych; – definiuje zmienne rekordowe; – podaje przykłady odwołania się do pola;	– deklaruje rekordy składające sie z różnych danych; – posługuje się deskryptorami pól; – omawia i stosuje strukturę tablicową zmiennych rekordowych; – uzasadnia stosowanie przez zmienną (var) w przypadku zmiany wartości zmiennej rekordowej wewnątrz funkcji lub procedury;	– samodzielnie planuje użycie struktur złożonych do łączenia w grupy danych, które mają być ze sobą powiązane, i wybiera te dane na podstawie analizy algorytmu;
Operacje wejścia i wyjścia – zapoznajemy się z plikami (27) Operacje wejścia i wyjścia – wykonujemy działania na plikach (28) Operacje wejścia i wyjścia – wykonujemy działania na plikach (29)	– omawia poszczególne etapy działań na plikach; – wie, jaka jest różnica pomiędzy plikami zdefiniowanymi a tekstowymi w Pascalu; – zna sens numerowania elementów w pliku;	– definiuje typy plikowe; – deklaruje zmienną plikową w bloku deklaracji; – kojarzy zmienną plikową z nazwą pliku; – wymienia i charakteryzuje etapy działań z plikiem;	– otwiera i tworzy pliki; – używa procedury write i writeln z parametrami (zmienną plikową, nazwą zmiennej); – używa procedury read readln z parametrami (zmienną plikową, nazwą zmiennej); – programuje pobranie, modyfikacje i zapis pliku; – prawidłowo zamyka plik procedurą close; – zapisuje dane w pliku za pomocą procedury write; – odczytuje plik procedurą read;	– posługuje się procedurą seek i funkcjami filesize, oef, eoln; – świadomie stosuje dwa sposoby dostępu do pliku – sekwencyjny i swobodny; – prawidłowo dobiera i wykorzystuje procedury i funkcje działające na plikach; – świadomie wykorzystuje dyrektywę kompilatora {$I};	– samodzielnie planuje wykorzystanie odpowiednich procedur działających na plikach do realizacji programów na podstawie analizy algorytmów;
Wykrywamy błędy (debugowanie) (30)	– zna istotę debugowania i jego znaczenie w pracy	– omawia różnicę pomiędzy opcjami debugowania (F7 i F8);	– wykorzystuje debugowanie do analizy błędów w	– uruchamia program metoda krokową, kontrolując zawartości	– biegle korzysta z debugera, analizując tylko wybrane

	programisty; – podaje przykłady, w których wskazane jest użycie debugera i uruchamiania krokowego programu; – wie, czym jest pułapka;	– debuguje prosty program i wskazuje miejsca wyświetlania danych debugowania, np. wartości zmiennych; – świadomie stosuje debugowanie do wykrywania błędów w krótkim programie bez procedur i funkcji;	programie z wywołaniem procedur i funkcji z niewielką ilością zmiennych; – umie kontrolować wartości zmiennych w trakcie debugowania; – uruchamia program metoda krokową, kontrolując zawartości zmiennych; – umie uruchomić debugowanie od dowolnego miejsca programu; – debuguje lub pomija w tym procesie podprogramy;	zmiennych; – umie uruchomić debugowanie od dowolnego miejsca programu; – debuguje lub pomija w tym procesie podprogramy; – odnajduje błędy za pomocą debugera lub pracy krokowej, porównując wartości zmiennych z przewidywanymi przez algorytm; – umie kontrolować wartości zmiennych w trakcie debugowania;	fragmenty programu. – typuje miejsca w programie, w których należy szukać błędu, i kontroluje je debugerem;
Rekurencja – wywołujemy samych siebie, „dziel i zwyciężaj” po raz pierwszy (31)	– omawia znaczenie struktury drzewa katalogowego i poddrzewa; – wie, na czym polega zasada rozwiązywania problemu „dziel i zwyciężaj”; – zna pojęcie rekurencji;	– zna i definiuje pojęcia autowywoływania podprogramu, rekurencji i iteracji, podaje przykłady; – wie, jakie rodzaje zmiennych powstają w czasie wywoływania procedury lub funkcji; – wyjaśnia pojęcie poziomu zagłębienia w procesach rekurencyjnych;	– układa program z zastosowaniem iteracji, obliczający silnię, i dokładnie objaśnia jego działanie jako funkcji; – wyjaśnia zasadę dostępu do poszczególnych zestawów zmiennych w procesach wykorzystujących rekurencję; – wie, na czym polega przeglądanie drzewa katalogowego metodą „dziel i zwyciężaj”; – wie, na czym polega działanie podprogramu wywołującego siebie; – na podstawie podręcznika układa program obliczający	– układa program (funkcję) obliczający silnię metodą iteracyjną i rekurencyjną; – analizuje poprawność ułożonego programu do obliczania silni za pomocą debugera; – zna różnicę pomiędzy rozwiązanie rekurencyjnym a iteracyjnym i umie oszacować, które z nich będzie korzystniejsze;	– układa algorytm przewidujący zastosowanie rekurencji oraz na jego podstawie samodzielnie tworzy funkcję lub procedurę;

			silnię metodą rekurencji;		
Modularyzacja programu – grupujemy podprogramy (32)	– zna budowę i składnię modułu; – wie, w jakim celu grupuje sie podprogramy;	– charakteryzuje części publiczną, implementacyjną i inicjującą modułu; – kompiluje moduły; –opisuje kolejność wykonywania części inicjujących;	– układa programy korzystające z modułów; – umie wyjaśnić różnicę pomiędzy modułem w postaci źródłowej a modułem w postaci wynikowej;	– używa do kompilacji modułów opcji Build; – samodzielnie wyodrębnia i typuje części programów, z których należy utworzyć moduł; – przewiduje zastosowania tworzonych modułów w innych programach;	– tworzy bibliotekę modułów, których wykorzystanie ułatwi układanie następnych programów;
II. Dynamiczne struktury danych					
W świecie wskaźników (1) Deklarujemy zmienne wskaźnikowe (2) Tworzymy pierwsze zmienne dynamiczne (3) Co jeszcze powinieneś wiedzieć o wskaźnikach? (4)	– wie, jakie są mechanizmy odwoływania się do zmiennych dynamicznych za pomocą wskaźników; – podaje przykłady takich zastosowań typów wskaźnikowych;	– wskazuje różnice pomiędzy zmiennymi wskaźnikowymi statycznymi a dynamicznymi;	– posługuje się funkcjami assigned oraz dispose, adispose; – wie, jak działają powyższe funkcje i procedura; – analizuje gotowe przykłady deklaracji zmiennych wskaźnikowych; – deklaruje zmienne wskaźnikowe; – umie odwołać się do zmiennej dynamicznej za pomocą zmiennej wskaźnikowej	– umie analizować błędy powstałe podczas tworzenia programu ze zmiennymi dynamicznymi; – umie analizować błędy powstałe podczas tworzenia programu ze zmiennymi dynamicznymi; – wie, czym jest spowodowana utrata dostępu do zmiennej dynamicznej; – zna znaczenie stałej nil; – określa zastosowania zmiennych wskaźnikowych takie jak odwołanie do zmiennych dynamicznych, użycie w instrukcji przypisania, użycie w relacji, jako wynik funkcji;	– samodzielnie układa programy z zastosowaniem zmiennych dynamicznych; – podaje uzasadnienie użycia zmiennych dynamicznych we własnym rozwiązaniu problemu informatycznego;

Dynamiczne struktury danych (5) Dynamiczne struktury danych – zapis kodu (6)	– umie podać przykład zastosowania dynamicznej struktury danych;	– zna znaczenie rekordu w tworzeniu struktur dynamicznych jako zmiennej dynamicznej; – wie, czym jest zmienna wskaźnikowa i umie opisać jej znaczenie w dynamicznej strukturze danych; – wie, czym jest pole wskaźnikowe w rekordzie;	– definiuje i deklaruje w programie struktury dynamiczne; – łączy zmienne dynamiczne na podstawie przykładów z podręcznika; – usuwa zmienną dynamiczną; – zmienia wartość zmiennej dynamicznej;	– samodzielnie łączy zmienne dynamiczne; – używa zmiennej wskaźnikowej do wypełniania pola rekordu dynamicznego; – wypełnia pola rekordu dynamicznego;	– samodzielnie układa programy z zastosowaniem struktur dynamicznych; – podaje uzasadnienie użycia struktur dynamicznych we własnym rozwiązaniu problemu informatycznego;
Stos – ostatni wchodzi, pierwszy wychodzi LIFO (7) Kolejka – pierwszy wchodzi, pierwszy wychodzi (FIFO) (8)	– umie opisać istotę stosu; – umie podać różnice pomiędzy LIFO a FIFO;	– zna budowę stosu i jego zastosowanie; – opisuje rolę wskaźników i zmiennych wskaźnikowych w tworzeniu i obsłudze kolejek;	– wie, jak zbudować stos dla zmiennych dynamicznych; – wie, jak przeglądać stos i jak usuwać zmienne dynamiczne; – wie, jak tworzyć w swoich programach kolejkę zmiennych; – wie; na czym polega wstawianie elementów do kolejki, jej przeglądanie i usuwanie elementów;	– samodzielnie buduje stos dla zmiennych dynamicznych; – zna funkcję zmiennej wskaźnikowej w adresowaniu wierzchołka stosu; – samodzielnie tworzy program przeglądający stos; – samodzielnie tworzy program usuwający zmienne dynamiczne; – tworzy w swoich programach kolejkę zmiennych;	– wie, jak zachowuje się komputer, gdy zostaje wywołane przerwanie, i wie, jaka w tym rola stosu procesora; – stosuje w swoich programach stos programowy; – identyfikuje w algorytmach i problemach informatycznych zagadnienia, w których należy zastosować FIFO lub LIFO;
Lista jednokierunkowa (9) Lista dwukierunkowa, cykliczna (10)	– wskazuje różnicę pomiędzy listą jedno- i dwukierunkową; – zna dokładnie różnice pomiędzy kolejką a stosem;	– wymienia cechy listy; – analizuje graficzne, podręcznikowe przedstawienie działań na listach; – wymienia niektóre problemy	– umie porządkować listy jednokierunkowe; – umie definiować w programie listy dwukierunkowe; – na podstawie podręcznika analizuje przykład programu z	– wstawia i usuwa elementy z listy jedno- i dwukierunkowej; – z powodzeniem stosuje w programach listy w tym także cykliczne;	– identyfikuje w algorytmach zagadnienia, które mogą być zrealizowane z wykorzystaniem list;

		informatyczne, w których stosuje się listy;	listą cykliczną; – definiuje rekord dla listy dwukierunkowej; – wie, czym jest i w jakich przypadkach znajduje zastosowanie lista cykliczna i podaje zasadniczą cechę listy cyklicznej; – samodzielnie analizuje kod programu, w którym zastosowano listy;		
Drzewo (11)	– opisuje analogię struktury drzewiastej do struktury folderów w systemie operacyjnym;				
III Bazy danych					
Tabele, wiersze i klucze (1)	– wie, czym jest baza danych; – zna znaczenie tabel, wierszy i kluczy dla bazy danych;	– wie, jakie znaczenie ma prawidłowe zaplanowanie i projektowanie tabel w kontekście poprawności i szybkości działania bazy danych; – wybiera odpowiednie nazwy dla pól tabeli; – operuje słownictwem znamiennym dla baz danych takim jak rekordy, pola, klucze;	– wskazuje pola, które jednoznacznie identyfikują rekordy, np. PESEL, numer telefonu itp.; – prawidłowo ustala klucze i identyfikatory dla tabel; – zna budowę rekordu tabeli; – wie, czym są klucze, klucze główne i czym się one różnią;	– trafnie dobiera pola do tworzenia kluczy zgodnie z założeniami projektowymi; – zna zastosowanie pól sztucznych autonumerowanych; – buduje tabele bazy zgodnie z założeniami projektowymi;	– samodzielnie projektuje rozbudowane tabele dla baz danych z jednoczesnym funkcjonalnym wyborem pół kluczy;
Projektujemy bazę danych. Pierwsza i druga postać normalna. (2)	– rozumie pojęcie i sens normalizacji oraz zna jej główny cel;	– wskazuje różnice pomiędzy 1., 2., 3. i 4. postacią normalną;	– ustala klucz główny dla 1. postaci normalnej tabeli; – umie przeprowadzić proces normalizacji do 2. postaci normalnej; – omawia rolę klucza	– zna zależności pomiędzy polami niekluczowanymi a kluczem głównym; – wskazuje miejsca, w których następuje redundancja danych;	– przeprowadza normalizację tabeli do postaci 4. normalnej z pominięciem normalizacji do postaci 2. i 3.;
Projektujemy bazę danych. Trzecia i czwarta postać					

normalna. (3)			głównego w 2. postaci normalnej;	– przeprowadza normalizację kolejno do wszystkich postaci normalnych;	
Projektujemy bazę danych. Określamy relacje miedzy tabelami. (4)	– odróżnia relacje 1–1 od 1–n; – wie, czym są relacje;	– określa prawidłowe relacje pomiędzy tabelami, używając kluczy; – zna pojęcia klucza głównego i klucza obcego;	– umie określić integralność bazy danych na podstawie analizy tabel;	– wykazuje, że po poprawnym procesie normalizacji tabele są powiązane prawidłowymi relacjami;	– samodzielnie określa relacje pomiędzy tabelami na podstawie własnego projektu bazy;
Pierwsze chwile z bazą danych – programy do tworzenia baz danych (5)	– wymienia nazwy darmowych i komercyjnych programów do tworzenia baz danych, w tym OpenOffice.org Base, Libre Office.org Base, Access; – umie uruchomić program to tworzenia relacyjnych baz danych, np. OpenOffice.org Base i zna rozmieszczenie opcji menu;	– zna i omawia funkcję obiektów głównego ekranu programu OpenOffice Org Base;	– posługuje się kreatorem bazy danych; – wie, jakie pliki powstają podczas tworzenia bazy;	– zna nazwy obiektów na ekranie głównym kreatora i omawia och przeznaczenie;	
Tworzymy tabele (6) Tworzymy i modyfikujemy tabele (7)	– odnajduje i uruchamia kreatora tabel; – wie, do czego służy i co można przy jego pomocy osiągnąć; – tworzy tabelę z pomocą podręcznika i nauczyciela;	– umie uruchomić i posługiwać się kreatorem tabel programu do tworzenia baz danych, np. OpenOffice.org Base; – wykazuje małą samodzielność w tworzeniu tabel i myli pola;	– samodzielnie tworzy tabele z użyciem kreatora; – prawidłowo ustala typy pól w tabelach; – modyfikuje nazwy tabel, pól i typów pól; – zna nazwy i przeznaczenie poszczególnych pól kreatora tabel; – wprowadza dane do tabeli;	– samodzielnie tworzy tabele bez użycia kreatora; – ustala klucze dla tabel; – bezbłędnie stosuje zasady nadawania nazw pól w tabeli; – prawidłowo i samodzielnie ustala klucze, w tym główne;	– samodzielnie realizuje bazę danych według własnego projektu; – samodzielnie planuje tabele, relacje, układa kwerendy, organizuje wprowadzanie i wyprowadzanie danych z bazy; – samodzielnie testuje bazę;

			– prawidłowo ustala klucze, w tym główne;		
Indeksujemy i określamy relacji (8)	– wyjaśnia pojęcie indeksu głównego;	– wyjaśnia, w jakim celu wprowadza się indeksowanie i relacje między tabelami; – wie, że klucze główne są indeksowane automatycznie;	– samodzielnie tworzy indeksy z wykorzystaniem kreatora; – samodzielnie ustala i tworzy relacje;	– planuje powiązania i indeksy zgodnie z założeniami wyszukiwania danych w bazie; – samodzielnie tworzy indeksy bez kreatora;	
Budujemy kwerendy z kreatora (9) Budujemy kwerendy bez kreatora (10) Budujemy kwerendy za pomocą SQL (11)	– wyjaśnia pojęcie kwerenda i jej znaczenie dla baz danych; – wie, czym jest SQL;	– układa kwerendy dla prostej bazy danych; – wyjaśnia, czym różni się kwerenda szczegółowa od skróconej; – omawia działanie najważniejszych poleceń języka SQL;	– używa kreatora do formułowania kwerend; – używa odpowiednich słów do wypełniania pól kreatora; – wybiera pola w kreatorze kwerend; – ustala porządek sortowania; – ustala warunki przeszukiwania; – sprawdza poprawność kwerendy; – tworzy tabele za pomocą języka SQL; – organizuje wydruk danych z kwerendy za pomocą SQL;	– świadomie używa wszystkich opcji kreatora w czasie tworzenia kwerendy; – przeprowadza testy poprawności działania kwerendy; – korzysta z aliasów; – samodzielnie tworzy kwerendę dla jednej i większej ilości tabel ze wszystkimi jej cechami bez użycia kreatora; – posługuje się edytorem SQL z programu OpenOffice.org Base; – tworzy kwerendę za pomocą SQL i edytora z programu OpenOffice.org Base; – tworzy kwerendy z warunkiem za pomocą SQL;	
Kreujemy formularze (12)	– umie używać gotowych formularzy do wprowadzania danych;	– korzysta z najważniejszych opcji kreatora formularzy; – stosuje style kreatora dla formularzy;	– umiejętnie wybiera pola formularza; – prawidłowo wybiera tryb wprowadzania danych;	– definiuje podformularze;	

		– prawidłowo określa nazwę formularza;	– prawidłowo rozmieszcza formaty; – prawidłowo wybiera tryby wprowadzania danych;		
Drukujemy raporty (13)	– wie, do czego służą raporty; – wie, który kreator służy do organizowania wydruków raportów.	– dobiera odpowiedni wygląd raportu; – drukuje gotowe raporty; – wie, że raporty tworzy się na podstawie tabeli lub kwerendy.	– używa kreatora i wszystkich jego opcji do tworzenia raportów; – prawidłowo wybiera pola do raportu; – prawidłowo i zgodnie z charakterem danych nazywa pola; – grupuje dane w raporcie.	– organizuje sortowanie danych w raporcie; – odróżnia raporty statyczne od dynamicznych.	

UWAGA! Do uzyskania oceny celującej w niektórych tematach wymagane są informacje i umiejętności wykraczające poza treść podręcznika