

Piotr Broda  
Danuta Smołucha

# INFORMATYKA

**PROGRAM NAUCZANIA  
DLA LICEUM OGÓLNOKSZTAŁCĄCEGO**

Program dopuszczony do użytku szkolnego przez Ministra Edukacji Narodowej i Sportu i wpisany do wykazu programów nauczania informatyki w klasach I–III liceum ogólnokształcącego w zakresie rozszerzonym na podstawie opinii rzeczoznawców mgr. inż. Krzysztofa Szota (z rekomendacji Polskiego Towarzystwa Informatycznego) oraz mgr. Marka Sadowskiego (z rekomendacji Ministra Edukacji Narodowej i Sportu).

**Numer dopuszczenia DKOS-5002-17/05**



Gdynia 2006

ISBN 83-7461-112-X

## Spis treści

<b>I. Wstęp – charakterystyka programu</b> .....	3
<b>II. Podstawa programowa</b> .....	5
<b>III. Szczegółowe cele edukacyjne i wychowawcze</b> .....	6
1. Cele edukacyjne .....	6
Algorytmika i programowanie .....	6
Bazodanowe funkcje arkusza kalkulacyjnego .....	7
Przetwarzanie informacji w relacyjnych bazach danych .....	8
Systemy i sieci komputerowe .....	8
Multimedia .....	9
Tendencje w rozwoju informatyki i jej zastosowań .....	9
2. Cele wychowawcze .....	9
<b>IV. Procedury osiągania celów</b> .....	10
1. Osiąganie celów edukacyjnych .....	10
2. Osiąganie celów wychowawczych .....	12
<b>V. Zadania szkoły</b> .....	13
Wyposażenie pracowni szkolnej .....	13
<b>VI. Ramowy rozkład materiału – proponowany przydział godzin</b> .....	14
<b>VII. Treści nauczania i przewidywane osiągnięcia ucznia</b> .....	15
Algorytmika i programowanie .....	15
Bazodanowe funkcje arkusza kalkulacyjnego .....	19
Przetwarzanie informacji w relacyjnych bazach danych .....	20
Systemy i sieci komputerowe .....	21
Multimedia .....	22
Tendencje w rozwoju informatyki i jej zastosowań .....	23
<b>VIII. Proponowany rozkład materiału</b> .....	24
Klasa druga .....	24
Klasa trzecia .....	27
<b>IX. Kontrola i ocena osiągnięć ucznia</b> .....	29
Klasyfikacja wymagań .....	29
Propozycja oceny według klasyfikacji wymagań .....	30
Metody oceniania uczniów .....	36
<b>X. Wymagania egzaminacyjne</b> .....	38
I. Wiadomości i rozumienie .....	39
II. Korzystanie z informacji .....	39
III. Tworzenie informacji .....	40

## I. Wstęp – charakterystyka programu

Informatyka jest dyscypliną naukową rozwijającą się w bardzo szybkim tempie, dlatego też program nauczania tworzony dla szkół ponadgimnazjalnych musi uwzględniać możliwość zarówno wprowadzania nowych treści, jak i modyfikacji oraz uzupełniania treści już nauczanych. Zwłaszcza zagadnienia związane z tendencjami rozwoju informatyki muszą być na bieżąco aktualizowane.

W prezentowanym programie ujęliśmy cele i treści nauczania zawarte w podstawie programowej przedmiotu informatyka, realizowanego w zakresie rozszerzonym, oraz standardy wymagań egzaminacyjnych z informatyki, które weszły w życie zgodnie z rozporządzeniem Ministra Edukacji Narodowej i Sportu z dnia 10 kwietnia 2003 roku, zmieniającym rozporządzenie w sprawie standardów wymagań będących podstawą przeprowadzania sprawdzianów i egzaminów.

W szczególności wzięliśmy pod uwagę fakt, że kształcenie na poziomie rozszerzonym w liceum powinno wpłynąć na decyzję uczniów o kontynuowaniu nauki na studiach wyższych i pomóc im w wyborze kierunku studiów. Duży nacisk kładziemy na przedstawienie uczniom podstaw dyscyplin związanych z informatyką, takich jak algorytmika i programowanie w językach wyższego poziomu. Zwracamy też szczególną uwagę na znajomość pojęć dotyczących sieci komputerowych oraz praktyczną umiejętność korzystania z narzędzi informatycznych przy tworzeniu baz danych i prezentacji multimedialnych. Algorytmikę i programowanie wprowadzamy do programu od podstaw, natomiast pozostałe zagadnienia były już realizowane na przedmiocie technologia informacyjna, zatem przewidujemy tylko rozszerzenie wiedzy i umiejętności ucznia.

Dodatkowo umieściliśmy w programie podstawy obsługi systemu Linux, jako że zakłada się dowolność narzędzi stosowanych przez ucznia na egzaminie maturalnym, pozostawiając również decyzji ucznia wybór platformy, na której będzie pracował. Dlatego uznaliśmy za wskazane zapoznać ucznia z systemem operacyjnym alternatywnym do powszechnie stosowanego.

Materiał realizowany w trakcie nauczania informatyki powinien być tak dobrany, by zachęcić ucznia do dalszego zgłębiania wiedzy samodzielnie bądź na wybranym kierunku studiów. Nie może zawierać nadmiaru treści, co zamiast inspirować, mogłoby działać w sposób zniechęcający. Należy zwrócić uwagę nie tylko na nabycie przez ucznia praktycznych umiejętności, ale również na wiedzę teoretyczną, zgodnie z wymogami standardów egzaminacyjnych.

Obok realizacji założeń podstawy programowej i standardów wymagań nasz program ma spełniać dodatkowe zadania:

- pomoc nauczycielowi w wyrównaniu różnic w umiejętnościach uczniów, którzy rozpoczęli naukę w liceum po szkołach o zróżnicowanym poziomie nauczania informatyki w cyklu kształcenia gimnazjalnego, oraz w takim rozplanowaniu i zorganizowaniu pracy, aby różnice te były jak najmniej widoczne i nie przesądzały o powodzeniu bądź jego braku w uczeniu się przedmiotu;

- wdrożenie ucznia do samodzielnej pracy, począwszy od poprawnej analizy problemu, poprzez odpowiedni dobór narzędzi i środków informatycznych, aż do realizacji i optymalnego rozwiązania;
- wdrożenie uczniów do pracy zespołowej, która jest wymagana przy realizacji niektórych projektów informatycznych; praca w grupie spełnia również rolę wychowawczą, przygotowując ucznia do wydajnej pracy zespołowej w przyszłości; uczeń ma świadomość, że efekty jego pracy i stopień zaangażowania mają wpływ na całokształt uzyskanego przez zespół wyniku;
- zrealizowanie możliwie w jak najszerszym stopniu treści dodatkowych w pracy z młodzieżą zarówno deklarującą chęć przystąpienia do egzaminu maturalnego z przedmiotu, jak i tą, która tej chęci początkowo nie wyraża (tu należy mieć świadomość, że w dużym stopniu od nas – nauczycieli – zależy, czy młodzież wybierze informatykę jako przedmiot maturalny);
- ustanowienie korelacji pomiędzy informatyką a innymi przedmiotami – szczególnie duże możliwości daje powiązanie: algorytmiki i programowania z matematyką, tworzenia arkuszy kalkulacyjnych z matematyką i fizyką, prezentacji multimedialnych z treściami realizowanymi na innym przedmiocie itp. (dużo zależy tu od współpracy nauczyciela informatyki z uczącymi innych przedmiotów);
- zaszczepienie w uczniu zasad etyki związanych z działaniem i funkcjonowaniem społeczeństwa informatycznego.

Obok treści nauczania proponujemy również przykładowy rozkład materiału skonstruowany dla pięciogodzinnego cyklu kształcenia w trzyletnim liceum ogólnokształcącym, dla klas o profilu matematyczno-informatycznym lub matematyczno-fizycznym, przy siatce godzin  $3 + 2$ , czyli trzy godziny lekcyjne w klasie drugiej oraz dwie godziny w klasie trzeciej.

Zakładając, że rok szkolny to pełne 36 tygodni nauki w klasie drugiej oraz 29 tygodni nauki w klasie trzeciej, i mnożąc tę liczbę przez trzy godziny tygodniowo w klasie drugiej oraz dwie godziny tygodniowo w klasie trzeciej, otrzymujemy  $108 + 58 = 166$  jednostek lekcyjnych. W propozycji rozkładu materiału, którą umieściliśmy w rozdziale VIII, przewidzieliśmy 90 i 50 godzin na realizację programu oraz odpowiednio 18 i 8 godzin do dyspozycji nauczyciela.

**Wyróżnioną czcionką** zaznaczyliśmy treści, które mogą zostać pominięte jako będące rozszerzeniem wiedzy i umiejętności ucznia. Decyzję dotyczącą sposobu zagospodarowania godzin przeznaczonych do dyspozycji nauczyciela pozostawiamy uczącemu, który oceniając stan wiedzy i umiejętności uczniów, zrealizuje treści dodatkowe lub przeznaczy je na utrwalanie umiejętności już zdobytych.

W naszym programie położyliśmy głównie nacisk na treści wskazane w standardach wymagań egzaminacyjnych, jako że ich realizacja zadecyduje o powodzeniu uczniów na egzaminie maturalnym. Treści te stanowią równocześnie wystarczającą podstawę do dalszego samodzielnego zgłębiania wiedzy przez ucznia. W pracy z uczniami zdolniejszymi, bardziej zainteresowanymi przedmiotem, proponujemy wprowadzenie treści dodatkowych, wykraczających poza program.

Tworząc program, zwracaliśmy największą uwagę na dwa aspekty: po pierwsze, aby umożliwić gruntowne przygotowanie ucznia do egzaminu maturalnego, a po drugie, aby był realny, czyli zawierał tyle treści, ile nauczyciel przy określonej liczbie godzin jest w stanie zrealizować.

## **II. Podstawa programowa**

Poniżej przedstawiamy dosłowne brzmienie podstawy programowej dla liceum ogólnokształcącego z informatyki. W dalszych rozdziałach będziemy się do niej odnosić, formułując tematy i zagadnienia umieszczone w programie.

### **Cele edukacyjne**

1. Przygotowanie do świadomego wyboru kierunku i zakresu dalszego kształcenia informatycznego.
2. Zdolność do samodzielnego korzystania z komputera dla realizacji części zadań edukacyjnych oraz innych celów poznawczych.

### **Zadania szkoły**

1. Stworzenie warunków do poznania wybranych zagadnień, pojęć i metod informatyki jako dyscypliny naukowej oraz jej najważniejszych zastosowań.
2. Kształcenie samodzielności intelektualnej, odpowiedzialności za własny rozwój, gotowości do podejmowania i rozwiązywania złożonych zadań, z uwzględnieniem środków i metod informatyki.
3. Rozwijanie umiejętności pracy zespołowej przez realizację projektów grupowych.

### **Treści nauczania**

1. Algorytmika i programowanie:
  - 1) metodyczna analiza i modelowanie umiarkowanie złożonych problemów i procesów z różnych dziedzin;
  - 2) przegląd algorytmów klasycznych;
  - 3) wybrane techniki projektowania algorytmów i struktur danych: programowanie strukturalne, zstępujące, abstrakcja danych, metoda kolejnych uściśleń;
  - 4) elementy analizy algorytmów;
  - 5) indywidualna i zespołowa realizacja projektów programistycznych w wybranym języku wysokiego poziomu.
2. Bazy danych:
  - 1) podstawowe formy organizacji informacji w bazach danych,
  - 2) budowa relacyjnych baz danych,
  - 3) wyszukiwanie informacji w relacyjnych bazach danych z użyciem języka zapytań,
  - 4) projektowanie prostych relacyjnych baz danych.

### 3. Multimedia. Sieci komputerowe:

- 1) sprawne i świadome korzystanie z multimediiów i tworzenie własnych materiałów multimedialnych,
- 2) przetwarzanie informacji w różnej postaci (w tym wizualnej i dźwiękowej),
- 3) budowa i działanie sieci komputerowych,
- 4) tworzenie i publikowanie własnych materiałów w sieci.

### 4. Tendencje w rozwoju informatyki i jej zastosowań.

#### **Osiągnięcia**

1. Formułowanie sytuacji problemowej, jej modelowanie i rozwiązywanie z użyciem metod informatycznych.
2. Ocenianie poprawności i efektywności rozwiązań i ich testowanie. Tworzenie dokumentów rozwiązań.
3. Wyszukiwanie informacji w bazach danych i projektowanie prostych baz danych.
4. Tworzenie opracowań multimedialnych.
5. Sprawne korzystanie z usług sieci komputerowych w pracy z informacjami swoimi i obcymi.
6. Planowanie pracy i nadzór nad przebiegiem wykonywania projektów realizowanych zespołowo z wykorzystaniem programów komputerowych.

## **III. Szczegółowe cele edukacyjne i wychowawcze**

Na lekcjach informatyki cele edukacyjne i wychowawcze powinny być realizowane w sposób spójny. Uczeń już od początku nauki przedmiotu musi znać aspekty etyczne związane z pracą z komputerem. W całym procesie dydaktycznym powinien mieć świadomość prawnych aspektów związanych z używanym oprogramowaniem. Uczeń musi w sposób przemyślany wykorzystywać informacje pozyskane w Internecie, uwzględniając aspekt poszanowania praw autorskich. Publikując w sieci własne materiały, powinien czuć się za nie odpowiedzialny.

Cele edukacyjne powinny być realizowane zgodnie z podstawą programową z uwzględnieniem maturalnego charakteru przedmiotu. Cele, które chcemy osiągnąć przy realizowaniu zagadnień nadprogramowych, zostały zaznaczone wyróżnioną czcionką.

### **1. Cele edukacyjne**

#### **Algorytmika i programowanie**

##### **Uczeń powinien:**

- znać sposób prezentowania danych w komputerze;
- rozumieć pojęcie zadania algorytmicznego, wskazywać zadania, które można zrealizować za pomocą algorytmu;
- znać i poprawnie tłumaczyć pojęcia związane z algorytmiką, takie jak poprawność algorytmu, optymalizacja, złożoność obliczeniowa, czasowa, pesymistyczna;

- umieć zapisać algorytm za pomocą pseudojęzyka, schematu blokowego, listy kroków, języka programowania;
- umieć zapisywać liczby w różnych systemach liczbowych;
- formułować zadanie algorytmiczne wraz z jego specyfikacją z wyszczególnieniem danych wejściowych i określeniem wyników;
- testować poprawność utworzonego przez siebie algorytmu;
- znać podstawowe metody programowania, takie jak programowanie liniowe, iteracja, programowanie z rozgałęzzeniami, rekurencja;
- kodować algorytmy w języku wysokiego poziomu, kompilować je i uruchamiać;
- dbać o przejrzystość pisanego kodu, stosować wcięcia i komentarze;
- dobrać odpowiednie typy danych do zadania algorytmicznego;
- porządkować zbiór – znać podstawowe metody sortowania: sortowanie bąbelkowe, przez wstawianie i selekcję oraz stosować je w implementacji programów;
- znać podstawowe metody numeryczne, takie jak przybliżanie pierwiastka kwadratowego liczby, rozwiązywanie równania kwadratowego, przybliżanie wartości liczby metodą Monte Carlo;
- znać pojęcie rekurencji, stosować rekurencję w prostych przykładach: obliczanie silni liczby naturalnej, znajdowanie potęgi całkowitej liczby rzeczywistej, znajdowanie  $n$ -tego wyrazu w ciągu Fibonacciego;
- rozwijać wielomian według schematu Hornera;
- stosować metodę „dziel i zwyciężaj” na podstawie przykładowych algorytmów;
- wskazywać na słabe strony rekurencji z uwzględnieniem złożoności pamięciowej i czasowej;
- sortować zbiór za pomocą metody szybkiej *quicksort*;
- testować napisane przez siebie programy dla różnych przypadków danych wejściowych;
- umiejętnie przeprowadzać dyskusję swojego rozwiązania z naciskiem na istotność zastosowanych środków i metod;
- realizować proste algorytmy liniowe i rozgałęzione w arkuszu kalkulacyjnym;
- wykonywać powierzone mu zadania w zespołowym projekcie programistycznym (np. napisać część funkcji w C++, które będą stanowiły integralną część całości projektu).

### Bazodanowe funkcje arkusza kalkulacyjnego

#### Uczeń powinien:

- sortować i filtrować dane w arkuszu (w tym stosować filtr zaawansowany);
- tworzyć wykresy różnego typu w zależności od prezentowanych na wykresie danych;
- sprawnie posługiwać się narzędziami importu i eksportu danych pomiędzy programami (np. import plików pomiędzy arkuszem kalkulacyjnym a edytorem tekstu);
- sporządzać podsumowania i zestawienia za pomocą sum pośrednich oraz tabel i wykresów przestawnych;
- uruchamiać istniejące makrodefinicje i pisać własne.

## **Przetwarzanie informacji w relacyjnych bazach danych**

### **Uczeń powinien:**

- wyszukiwać informacje w istniejącej bazie danych z zastosowaniem filtrów, kwerend i raportów;
- znać schemat budowy relacyjnej bazy danych;
- projektować własne relacyjne bazy danych;
- stworzyć relacyjne bazy danych, umiejętnie łącząc pola tabel odpowiednimi relacjami;
- stosować kwerendy w utworzonej przez siebie bazie: wyszukującą, aktualizującą, parametryczną (oparte zarówno na jednej, jak i wielu tabelach połączonych relacjami);
- stworzyć własne bazy danych;
- poprawnie organizować dane w relacyjnej, wielotabelowej bazie danych;
- porządkować informacje w bazie;
- modyfikować dane w bazie;
- stosować mechanizmy importu/eksportu danych do bazy i z bazy;
- interpretować zapytania formułowane w języku SQL;
- umiejętnie posługiwać się poleceniami języka SQL;
- rozumieć i stosować mechanizmy ochrony danych w bazie.

## **Systemy i sieci komputerowe**

### **Uczeń powinien:**

- wiedzieć, czym są systemy operacyjne i umieć wymienić kilka z nich;
- znać historię systemu Linux i wiedzieć, jakie są tendencje w jego rozwoju;
- zalogować i wylogować się z systemu Linux oraz prawidłowo zamknąć system;
- przeglądać, kopiować i usuwać pliki oraz foldery w systemie Linux;
- interpretować i ustalać prawa dostępu w systemie Linux;
- montować urządzenia w systemie;
- przeglądać i przerywać działające procesy;
- efektywnie wykorzystywać środowiska graficzne systemu Linux (KDE, GNOME);
- korzystać z narzędzi technologii informacyjnej dostępnych w systemie Linux (pakiet biurowy, edytor grafiki) oraz wymieniać pliki pomiędzy systemami;
- wiedzieć, czym jest sieć komputerowa;
- znać pojęcia związane z budową lokalnej sieci komputerowej;
- znać podstawy prawne związane z tworzeniem i rozpowszechnianiem informacji w sieci lokalnej i globalnej;
- znać metody przesyłu przez sieć informacji i plików, porozumiewać się przez sieć z innymi użytkownikami w czasie rzeczywistym;
- wymienić i scharakteryzować typy serwerów;
- znać architekturę sieci lokalnych, opisywać urządzenia wspomagające pracę w sieci;
- rozumieć pojęcie protokołu sieciowego, znać zasadę budowy adresu IP;
- charakteryzować typowe narzędzia informatyczne i znać ich zastosowania.

## **Multimedia**

### **Uczeń powinien:**

- wymienić i scharakteryzować kilka przykładowych formatów plików dźwiękowych;
- znać różnice pomiędzy grafiką rastrową i wektorową;
- znać zasady tworzenia sceny 3D i generować proste sceny w wybranym programie;
- znać zasady tworzenia animacji, wykonywać własne animacje;
- wykorzystywać pliki dźwiękowe i graficzne do tworzenia własnych prezentacji WWW;
- wykorzystywać skrypty Java w kodzie HTML;
- umieć dokonać autoprezentacji w sieci za pomocą utworzonej własnej strony internetowej zapisanej w języku HTML.

### **Tendencje w rozwoju informatyki i jej zastosowań**

#### **Uczeń powinien:**

- znać nowe osiągnięcia i kierunki rozwoju technik informacyjnych i informatyki;
- wiedzieć, jak rozwija się przestrzeń zastosowań komputerów;
- znać kierunki rozwoju oprogramowania i systemów operacyjnych;
- mieć świadomość ograniczeń związanych z możliwościami komputerów;
- znać aspekty społeczne, prawne i ekonomiczne rozwoju Internetu.

## **2. Cele wychowawcze**

Cele wychowawcze, których realizację zakłada szkoła, to:

- przekonanie ucznia do konieczności przestrzegania regulaminu pracowni komputerowej i odpowiedzialnego użytkowania sprzętu w pracowni;
- wdrożenie ucznia do planowania pracy zarówno indywidualnej, jak i zespołowej;
- przygotowanie ucznia do świadomego wyboru kierunku i zakresu dalszego kształcenia informatycznego;
- wpojenie poszanowania własności intelektualnej innych osób, zwrócenie uwagi ucznia na etyczny aspekt przestrzegania prawa autorskiego w kontekście oprogramowania i publikacji w sieci;
- wykształcenie odpowiedzialności za treści prezentowane przez ucznia w sieci;
- dokonywanie selekcji i oceny treści dostępnych w sieci Internet, z uwzględnieniem ich wiarygodności i użyteczności;
- motywacja do znajdowania własnych rozwiązań, pobudzanie zainteresowania ucznia do pogłębiania zdobytej wiedzy;
- nauka korzystania z usług sieciowych z zachowaniem szacunku dla wszystkich użytkowników sieci (łącznie z użytkownikami anonimowymi);
- przekonanie o szkodliwości stron nacjonalistycznych, pornograficznych itp.;
- wskazanie korzyści i zagrożeń wpływających z zastosowania współczesnych osiągnięć technologii informacyjnych i informatyki.

## IV. Procedury osiągnięcia celów

### 1. Osiągnięcie celów edukacyjnych

Osiągnięcie założonych celów edukacyjnych zależy od wielu czynników. Spośród nich największy wpływ na jakość osiągniętych wyników mają metody nauczania. Powinny być one dobierane stosownie do zdolności i możliwości intelektualnych uczniów, a także odpowiednio do treści, które są realizowane.

Metody osiągnięcia celów edukacyjnych można podzielić na metody podające, takie jak wykład, opis, wyjaśnianie, oraz metody poszukujące, które zmuszają ucznia do aktywnego uczestnictwa w lekcji. Do tych drugich zaliczamy między innymi pogadankę, burzę mózgów, dyskusję, polemikę.

Realizacja osiągnięcia większości celów w przedmiocie informatyka wymaga zastosowania wielu metod równocześnie. Ze względu na specyfikę przedmiotu informatyka daje możliwość wykorzystania bardzo wielu środków służących realizacji zamierzonych celów edukacyjnych.

Na lekcjach informatyki pracuje się zwykle z małą liczbą uczniów, gdyż na przedmiocie tym obowiązuje podział liczniejszych klas na grupy. Daje to możliwość pełniejszej kontroli nauczyciela nad poczynaniami ucznia.

Wśród metod podających stosowanych przez uczącego na pierwszy plan wysuwa się wykład jako środek wprowadzający ucznia w temat lekcji. Można się posłużyć wykładem problemowym, opowiadaniem, opisem lub pokazem prezentacji. Na lekcjach informatyki wykład powinien być tylko poprzedzeniem ćwiczeń wykonywanych na komputerze. Nauczyciel przedstawia i wyjaśnia problem, który uczniowie będą rozwiązywać. Wykład powinien być tak skonstruowany, aby zachęcał ucznia do wyrażania własnych wniosków i propozycji dotyczących realizacji i rozwiązania postawionego problemu. Na pewno większą satysfakcję sprawi uczniowi samodzielne dochodzenie do rozwiązania aniżeli praca na zasadzie wykonywania sekwencji poleceń. Proponujemy, aby nauczyciel do każdego realizowanego zagadnienia problemowego przygotował zestaw wskazówek, które stopniowo będą ucznia przybliżać do rozwiązania. Nauczyciel powinien przedstawiać zadania do realizacji w taki sposób, aby uczniowie byli zmuszeni do samodzielnego poszukiwania.

Pożądane jest, aby uczniowie przedstawiali swoje propozycje oraz przeprowadzali wspólnie z nauczycielem dyskusje dotyczące poprawności możliwych rozwiązań. Nauczyciel powinien zachęcać ucznia do szukania przykładów związanych z realizowanymi zagadnieniami teoretycznymi oraz weryfikować ich trafność. Ciekawe pomysły na rozwiązanie problemu należy premiować ocenami, plusami bądź pochwałą.

Pomysły uczniów mogą być poddawane analizie i ocenie całej grupy. Warto od czasu do czasu poddać analizie i ocenie również pomysły nauczyciela. Wówczas uczniowie sprawdzają poprawność pomysłu, jego optymalność i przydatność w praktycznym zastosowaniu, w sposób jasny motywując swoją opinię. Pojawia się przy tym możliwość realizacji konstruktywnej krytyki, środka raczej rzadko stosowanego przy realizacji celów edukacyjnych, mającego jednak ogromne znaczenie nie tylko edukacyjne, ale i wychowawcze.

Przy niektórych ciekawych tematach nauczyciel może podjąć ryzyko konfrontacji uczniów z problemem, który wyjaśni, nie dając jednak żadnych wskazówek dotyczących rozwiązania. Można się wówczas spodziewać tak zwanej burzy mózgów, prowadzącej do ciekawych rozwiązań, a ponadto będącej metodą niezwykle aktywizującą uczniów.

Kolejnym etapem lekcji (po przedstawieniu i wyjaśnieniu zagadnienia przez nauczyciela, propozycji i weryfikacji rozwiązań przez uczniów i nauczyciela) jest realizacja rozwiązania. W tej części posłużymy się metodami praktycznymi, wśród których na ogół będzie ćwiczenie przy komputerze. Preferujemy raczej samodzielną pracę ucznia z komputerem.

Zawsze, jeśli jest to możliwe, należy zaproponować uczniom treść zadania w kilku wersjach o zróżnicowanym stopniu trudności. Dla przykładu, przy realizacji zadania: „Zaimplementuj program, który podaną przez użytkownika liczbę całkowitą zamienia na liczbę w postaci dwójkowej” uczniom zdolniejszym proponujemy treść: „Zaimplementuj program, który podaną przez użytkownika liczbę całkowitą zamienia na liczbę w systemie o podstawie  $n$ , gdzie  $n$  jest liczbą całkowitą z przedziału «2; 9»”. W ten sposób unikamy sytuacji, w której zadanie do realizacji jest zbyt trudne dla niektórych uczniów, a dla innych zbyt łatwe – w obu przypadkach część uczniów nie uczestniczyłaby aktywnie w lekcji.

Specyfika lekcji informatyki (optymalnie: jeden uczeń przy jednym komputerze) sprawia, że każdy uczeń musi sam wykonać własną pracę, po jej omówieniu i wyjaśnieniu ewentualnych dodatkowych wątpliwości i zapytań. Dlatego przed zakończeniem każdej lekcji nauczyciel powinien podsumować stopień i jakość wykonania zadania przez każdego ucznia. Ma to znaczenie motywujące – uczeń wie, że jego praca będzie przez nauczyciela oceniona, przy czym musi to być stopień wpisany do dziennika. Motywujący jest również fakt, że w przypadku realizacji poszczególnych tematów z informatyki uczeń sam w sposób wymierny jest w stanie ocenić wyniki swojej pracy. Dla przykładu, jeśli program zaimplementowany przez ucznia w języku programowania uruchamia się i działa w sposób poprawny, wówczas uczeń ma satysfakcję, że wykonał powierzone mu zadanie.

Praca w grupach przy projektach informatycznych nie tylko motywuje ucznia do brania odpowiedzialności za całą grupę, ale stwarza też możliwość porównywania własnych umiejętności z umiejętnościami pozostałych członków grupy.

Wymiana informacji między uczniami wspólnie realizującymi projekt grupowy, jak i konsultacja z nauczycielem przy pracy indywidualnej jest ważnym elementem osiągnięcia celów edukacyjnych. Uczeń, chcąc uzyskać radę, wskazówkę czy podpowiedź, musi się posługiwać poprawnym językiem informatycznym, na co nauczyciel powinien zwracać szczególną uwagę.

Bardzo ważnym elementem, mającym duży wpływ na zaangażowanie uczniów na lekcjach informatyki, jest **praktyczne wykorzystanie** umiejętności zdobytych na lekcjach. Dlatego warto wskazać uczniom sytuacje, w których aktualnie zdobywana wiedza ma praktyczne zastosowanie.

Kolejnym czynnikiem służącym osiągnięciu celów edukacyjnych jest systematyczne sprawdzanie wiedzy ucznia. Oceny z kartkówki i sprawdzianów oraz ocenianie bieżące są ważną informacją zwrotną dla samego ucznia o jego postępach w nauce. W rozdziale IX proponu-

jemy przeprowadzanie nie tylko sprawdzianów zapowiedzianych, ale również niezapowiedzianych kartkówek lub „komputerówek”, czyli krótkich sprawdzianów praktycznych. Motywuje to ucznia do systematycznej pracy.

Pamiętajmy, że tylko systematyczna praca daje uczniowi możliwość osiągnięcia wysokich wyników. Wyniki uzyskiwane w cyklicznie organizowanych sprawdzianach opartych na zasadach nowej matury mają działanie motywujące nie tylko dla potencjalnych maturzystów chcących zdawać ten przedmiot. Formuła arkusza z podziałem na zadania zamknięte i otwarte stanowi wyzwanie dla wszystkich uczniów, jest bowiem konfrontacją z nowatorską formułą sprawdzania wiedzy. Musimy przewidzieć sytuacje, w których pomimo starań nauczyciela uczeń niedokładnie zrozumie treści prezentowane na lekcji ze względów od nas niezależnych, na przykład z powodu absencji w szkole. Stąd ważna rola podręcznika, który nauczyciel na początku cyklu nauczania zaproponował uczniom i według którego realizuje program nauczania. Wskazane jest, aby nauczyciel na każdą lekcję przygotowywał materiały pomocnicze dla uczniów, które uczeń może pobrać w postaci elektronicznej. Rozwiązaniem optymalnym jest dodatkowa godzina konsultacji tygodniowo, podczas której nauczyciel pozostaje do dyspozycji uczniów chcących się podzielić swoimi uwagami lub proszących o wyjaśnienie niezrozumianych zagadnień.

## 2. Osiągnięcie celów wychowawczych

Zadania nauczyciela nie ograniczają się do przekazywania uczniom treści danego przedmiotu. Nauczyciel powinien podjąć wysiłek rozwijania i krystalizowania zainteresowań ucznia, a także pomóc mu uświadomić sobie zdolności i predyspozycje, tak aby ułatwić następnie wybór zawodowej kariery po ukończeniu liceum.

Prawidłowa konstrukcja lekcji pomaga nie tylko w osiągnięciu celów edukacyjnych, ale równocześnie pełni rolę wychowawczą. Indywidualizacja pracy uczy odpowiedzialności, planowania i organizacji własnej pracy. Motywuje ucznia do samodzielnego szukania rozwiązań, uczy prezentacji wyników swojej pracy.

Praca w grupach kształtuje odpowiedzialność za całą grupę, rozwija umiejętność podziału ról pomiędzy jej członków, wpaja solidność w wykonywaniu swojej części zadania. Branie udziału w dyskusji i burzy mózgowi doskonali umiejętność kulturalnej wymiany poglądów.

Od początku kształcenia informatycznego, już w klasie pierwszej, na technologii informacyjnej, należy zwracać szczególną uwagę uczniów na **poszanowanie własności intelektualnej**, zwłaszcza **przestrzeganie praw autorskich**. Sugerujemy, aby już przy implementacji prostych programów w języku programowania zalecać uczniom rozpoczynanie kodowania programu od informacji o tym, co program wykonuje i czyjego jest autorstwa.

Pamiętajmy również, że nauczyciel informatyki jest pedagogiem, który w całym cyklu kształcenia pełni rolę koordynatora poczynań ucznia. Zatem, od początku pracy z uczniem, kładzie nacisk na aspekt moralny, etyczny i prawny korzystania z oprogramowania licencjonowanego, zachęca ucznia do autoprezentacji w sieci i ostrzega przed bezmyślnym korzystaniem z zasobów sieci (informacje zaczerpnięte z Internetu mają być pomocą w szukaniu własnych rozwiązań, a nie kopiowaniem cudzych). Wskazanie przykładów nieprawdziwych lub niepełnych informacji zaczerpniętych z sieci pokazuje uczniowi konieczność weryfikacji

treści umieszczonych w Internecie (częste są strony typu „poradnik programisty”, które podają kody źródłowe zawierające dużą liczbę błędów).

Na lekcjach informatyki kontynuujemy egzekwowanie przestrzegania przez ucznia regulaminu pracowni oraz zwracamy uwagę na świadome i odpowiedzialne użytkowanie znajdującego się tam sprzętu.

## **V. Zadania szkoły**

Głównym zadaniem szkoły jest pomoc uczniom w osiągnięciu i doskonaleniu wiedzy i umiejętności, z uwzględnieniem realizacji wszystkich założonych celów edukacyjnych. Należy zwrócić szczególną uwagę na następujące zadania:

1. Stworzenie warunków do poznania wybranych zagadnień, pojęć i metod informatyki, jako dyscypliny naukowej oraz jej najważniejszych zastosowań.
2. Kształcenie samodzielności intelektualnej, odpowiedzialności za własny rozwój, gotowości do podejmowania i rozwiązywania złożonych zadań z uwzględnieniem środków i metod informatyki.
3. Rozwijanie umiejętności pracy zespołowej przez realizację projektów grupowych.
4. Rozwijanie umiejętności pracy indywidualnej, jej planowania i organizacji.

Szkoła ma obowiązek tak zorganizować pracownię informatyczną, aby praca w niej była bezpieczna i wydajna.

### **Wyposażenie pracowni szkolnej**

Zakładamy, że realizacja programu dla klas o rozszerzonym profilu nauczania informatyki powinna się odbywać w szkolnych pracowniach komputerowych, które zapewniają uczniom samodzielną pracę, optymalnie jeden uczeń przy jednym stanowisku komputerowym. Stanowi to konieczny warunek do pełnej aktywizacji każdego ucznia na lekcji, indywidualizacja pracy sprawia, że uczeń jest świadomy pełnej odpowiedzialności za wykonywane przez siebie zadania. Komputery należy połączyć w sieć lokalną, a stacje powinny mieć dostęp do sieci Internet, w celu realizacji tematów z nią związanych. Polecamy przydzielenie każdemu uczniowi określonej przestrzeni dyskowej w postaci folderu, w którym zapisuje efekty swojej pracy na lekcjach. Dostęp do zasobów uczniowskich powinien mieć tylko sam uczeń za pomocą uwierzytelniającego hasła oraz nauczyciel (prawo odczytu).

Poniżej przedstawiamy propozycję oprogramowania stacji komputerowych w odniesieniu do treści realizowanych zgodnie z programem nauczania.

Dział programowy	Minimalny proponowany zestaw oprogramowania	Uwagi
Algorytmika i programowanie	Delphi 7.0, jeśli wybranym językiem programowania jest Pascal Builder 6.0 lub Dev C++, jeśli wybranym językiem programowania jest C/C++	Wybór języka programowania zależy od nauczyciela, nie ma bowiem czasu w dwuletnim cyklu kształcenia na naukę obydwu języków. Dopuszczalne jest stosowanie dowolnego kompilatora Pascala i C/C++.
Przetwarzanie informacji w relacyjnych bazach danych	Pakiet Microsoft Office – Access	Wyboru dokonano ze względu na powszechność stosowania. Istotną jest możliwość eksportu plików pomiędzy składnikami pakietu.
Arkusze kalkulacyjne	Pakiet Microsoft Office – Excel	
Systemy i sieci komputerowe	Przeglądarka stron WWW, na przykład Internet Explorer, system operacyjny Linux dowolnej dystrybucji	W dziale „Sieci komputerowe” część realizowanych tematów oparta jest na wykładzie nauczyciela i poszukiwaniu w Internecie informacji dotyczących poznawanego materiału.
Multimedia	PowerPoint do tworzenia prezentacji multimedialnych GIMP – program do tworzenia grafiki, klient FTP do publikacji stron w Internecie, IrfanView, POV-Ray – grafika 3D	

## VI. Ramowy rozkład materiału – proponowany przydział godzin

Uczeń w klasie z rozszerzonym zakresem informatyki rozpoczyna naukę przedmiotu po uprzednim zaliczeniu w klasie pierwszej technologii informacyjnej. Dlatego też niektóre treści nauczanego przedmiotu są kontynuacją wiedzy zdobytej wcześniej.

Nowym dziełem jest algorytmika i programowanie. Na jego realizację przeznaczamy zdecydowanie największą liczbę godzin, mianowicie całość godzin przeznaczonych dla klasy drugiej. Oczywiście od nauczyciela zależy, czy 18 godzin, które proponujemy na zapoznanie ucznia z zaawansowanymi technikami programowania, przeznaczy na realizację naszej propozycji czy też kolejnych treści programowych zamieszczonych przez nas w klasie trzeciej.

W dwuletnim cyklu kształcenia zakładamy w klasie drugiej liczbę godzin 90 + 18, a w klasie trzeciej 50 + 8 i proponujemy następujący rozkład materiału:

Algorytmika i programowanie	90 godzin
Przetwarzanie informacji w relacyjnych bazach danych	11 godzin
Systemy i sieci komputerowe, multimedia	27 godzin
Tendencje w rozwoju informatyki i jej zastosowań	4 godziny
Ćwiczenia z arkuszami maturalnymi	8 godzin

Ćwiczenia z arkuszami maturalnymi pełnią rolę powtórzenia całości materiału z obu klas i przygotowują do egzaminu maturalnego. Zakładamy, że praca z arkuszami może być realizowana jako praca indywidualna ucznia lub też praca grupowa, polegająca na wspólnym rozwiązywaniu zadań umieszczonych w arkuszu. W szczegółowym rozkładzie materiału nie umieściliśmy tych ćwiczeń w konkretnym czasie, pozostawiając decyzję nauczycielowi.

## VII. Treści nauczania i przewidywane osiągnięcia ucznia

Treści nauczania oparte zostały na standardach wymagań egzaminacyjnych obowiązujących na maturze z informatyki, począwszy od matury 2005 roku, oraz podstawie programowej przedmiotu. W poniższej tabeli będziemy się odnosić bezpośrednio do wymagań egzaminacyjnych zgodnych z założeniami nowej matury, które umieszczone są w rozdziale X.

### Algorytmika i programowanie

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
<b>Algorytmika</b>		
Reprezentacja danych w komputerze – systemy zapisu liczb.	<ul style="list-style-type: none"> <li>– zna definicje bitu, bajtu oraz pojęcie systemu pozycyjnego</li> <li>– potrafi przeliczać liczby zapisane w dowolnym systemie liczbowym na inny system</li> </ul>	I. 1.
Pojęcie algorytmu, formułowanie sytuacji i zjawisk, które da się przedstawić za pomocą algorytmu.	<ul style="list-style-type: none"> <li>– umie powiązać pewne sytuacje życiowe z algorytmem</li> <li>– potrafi wskazać przykłady algorytmów w materiale nauczania innych przedmiotów (np. matematyki, fizyki) i w życiu codziennym</li> </ul>	I. 1. II. 5. III. 1.
Określenie specyfikacji problemu, wyróżnienie danych wejściowych oraz wyjściowych otrzymywanych w wyniku działania algorytmu. Wyszczególnienie kolejnych kroków działania algorytmu.	<ul style="list-style-type: none"> <li>– dla klasycznych przypadków algorytmów wyszczególnia kolejne etapy algorytmu, identyfikuje dane wejściowe, wyjściowe, zmienne pomocnicze</li> <li>– poprawnie zapisuje specyfikację problemu algorytmicznego</li> </ul>	I. 1. II. 7. III. 1. III. 2. III. 3. III. 4.
Zapis algorytmu za pomocą: – pseudojęzyka, z wyszczególnieniem listy kroków, – schematu blokowego.	<ul style="list-style-type: none"> <li>– umie zapisać algorytm</li> <li>– wybiera najkorzystniejszą formę zapisu ze względu na jakość algorytmu i problem, który algorytm rozwiązuje</li> </ul>	I. 1. II. 5. II. 7. III. 1. III. 2.
Pojęcia związane z algorytmem: poprawność, skończoność, złożoność algorytmu, uniwersalność. Analiza porównawcza różnych algorytmów rozwiązujących ten sam problem.	<ul style="list-style-type: none"> <li>– oblicza złożoność algorytmów dla prostych przykładów</li> <li>– wybiera algorytm najbardziej optymalny</li> <li>– umie ocenić poprawność algorytmu, jego zgodność ze specyfikacją problemu</li> </ul>	I. 1. II. 5. II. 7. III. 1. III. 4.
Analiza schematów blokowych algorytmów, odczytywanie sposobu działania na podstawie schematu blokowego. Odczytywanie danych wyjściowych algorytmu dla zadanych danych wejściowych.	<ul style="list-style-type: none"> <li>– odczytuje i nazywa problem na podstawie schematu blokowego</li> <li>– dla zadanych danych wejściowych potrafi przez analizę odczytać zbiór wyników działania algorytmu</li> </ul>	I. 1. II. 7. II. 8. III. 4.

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
<b>Programowanie</b>		
Język programowania. Klasyfikacja języków programowania. Wybór języka programowania, w którym uczniowie będą zapisywać algorytmy, i uzasadnienie tego wyboru.	<ul style="list-style-type: none"> <li>– umie wymienić kilka podstawowych języków programowania</li> <li>– wie, na czym polega różnica pomiędzy językami niższego i wyższego poziomu</li> </ul>	I. 1. I. 3. II. 1.
Wyróżnienie etapów programowania: specyfikacja problemu, zapis za pomocą schematu blokowego, implementacja w języku programowania, kompilacja, uruchomienie, testowanie przypadków granicznych. Rola kompilatora, linkera, debugera. Podstawowe błędy kompilacji i ich likwidowanie. Konstruowanie kodu źródłowego z zachowaniem estetyki i czytelności (wcięcia, komentarze).	<ul style="list-style-type: none"> <li>– zna podstawowe pojęcia związane z programowaniem w danym języku</li> <li>– zna rolę kompilatora, debugera, linkera</li> <li>– umie wyjaśnić pojęcie kompilacji</li> <li>– rozumie potrzebę dołączania plików nagłówkowych</li> <li>– rozumie rolę pamięci operacyjnej i procesora w działaniu programu</li> <li>– ma świadomość potrzeby zachowania zasad estetyki tworzonego przez siebie kodu i stosuje je w praktyce</li> </ul>	I. 1. II. 1. II. 7. II. 8. III. 4.
Stałe i zmienne w programie. Podstawowe typy zmiennych – typy proste. Nazewnictwo stałych i zmiennych, słowa kluczowe.	<ul style="list-style-type: none"> <li>– potrafi wymienić podstawowe typy zmiennych</li> <li>– umie dla danej sytuacji problemowej określić potrzebę zastosowania zmiennych wybranego przez siebie typu</li> <li>– zna potrzebę zastosowania stałych w programie</li> </ul>	III. 1. III. 2. III. 3.
Instrukcje: przypisania i porównania.	<ul style="list-style-type: none"> <li>– umie zastosować instrukcje przypisania i porównania</li> <li>– potrafi rozróżnić sytuacje ze względu na zastosowanie obu instrukcji</li> </ul>	III. 1. III. 2. III. 3.
Podstawowe operatory: matematyczne, relacji i logiczne.	– umie zastosować podstawowe operatory w celu rozwiązania problemu algorytmicznego	III. 1.
Instrukcja warunkowa i instrukcja wyboru, zastosowanie w programach (np. rozwiązywanie równania liniowego, równania kwadratowego).	– stosuje instrukcje warunkowe i wyboru przy rozwiązywaniu prostych problemów algorytmicznych	I. 5. a II. 7. II. 8. III. 1. III. 2. III. 4.
Implementacja prostych algorytmów liniowych i rozgałęzionych w arkuszu kalkulacyjnym.	– umie tworzyć arkusze kalkulacyjne, rozwiązujące proste problemy algorytmiczne	I. 5. II. 1 III. 1
Algorytmy z instrukcjami pętli, algorytmy iteracyjne (np. szukanie miejsca zerowego funkcji w zadanym przedziale za pomocą metody kolejnych przybliżeń, algorytm znajdowania losowo wygenerowanej liczby z zadanego przedziału), zastosowanie generatora liczb losowych (np. metoda Monte Carlo, ruchy Browna). Algorytm badania, czy liczba jest pierwsza, algorytm znajdowania największe-	<ul style="list-style-type: none"> <li>– umie stosować podstawowe instrukcje języka przy rozwiązywaniu problemów numerycznych</li> <li>– rozumie i stosuje metody iteracyjne oraz metody kolejnych przybliżeń</li> <li>– ma świadomość istnienia generatora liczb pseudolosowych i stosuje go w implementowanych przez siebie programach</li> </ul>	I. 1. I. 5. a II. 5. II. 7. II. 8. III. 1. III. 2. III. 4. III. 6.

go wspólnego dzielnika dla pary liczb. Znajdowanie pierwiastka kwadratowego z liczby dodatniej metodą Newtona-Raphsona.	<ul style="list-style-type: none"> <li>– zapoznaje się z klasycznymi algorytmami iteracyjnymi</li> <li>– wie, czym są metody numeryczne, pozna je na przykładach</li> </ul>	
Procedury i funkcje. Deklaracja, sposób przekazywania parametrów do funkcji: parametry formalne i aktualne. Zmienne globalne i lokalne.	<ul style="list-style-type: none"> <li>– rozumie potrzebę tworzenia autonomicznych i uniwersalnych funkcji w programie, tworzy je i potrafi wykorzystać</li> <li>– zna sposoby przekazywania argumentów do funkcji i umie je dopasowywać do rozwiązywanego problemu</li> </ul>	<p>II. 8. III. 2. III. 4.</p>
<b>Całka Riemanna – implementacja programu obliczającego całkę oznaczoną bezpośrednio z definicji.</b>	<ul style="list-style-type: none"> <li>– potrafi powiązać wiedzę matematyczną z umiejętnością implementacji programów</li> <li>– korzystając z metody definicyjnej całki Riemanna, oblicza pole powierzchni ograniczonej wykresem funkcji</li> <li>– stosuje poznane metody i środki do napisania programu liczącego wartość całki na podstawie definicji</li> </ul>	<p>III. 1. III. 2. III. 3. III. 4. III. 6. III. 7.</p>
Typy tablicowe – tablica jedno- i wielowymiarowa, sposób przekazywania tablic do funkcji. Wykorzystanie tablic do implementacji klasycznych algorytmów, np. zamiana zapisu liczby z systemu dziesiętnego na system binarny.	<ul style="list-style-type: none"> <li>– rozumie sposób działania i obsługi tablic</li> <li>– umie w sposób poprawny stosować tablice w programach</li> <li>– zapisuje liczbę w systemie binarnym lub systemie o innej podstawie liczenia (np. w systemie trójkowym, czwórkowym itp.)</li> </ul>	<p>II. 5. II. 7. III. 2.</p>
Przeszukiwanie tablicy w celu znalezienia wyróżnionego elementu, znajdowanie powtarzających się elementów tablicy. Metoda „dziel i zwyciężaj” – zastosowanie w pracy z tablicą jednowymiarową.	<ul style="list-style-type: none"> <li>– umie obsługiwać tablicę jednowymiarową</li> <li>– porównuje algorytmy rozwiązujące ten sam problem (np. przeszukiwanie tablicy z wartownikami i bez)</li> <li>– poznaje, na czym polega metoda „dziel i zwyciężaj”</li> </ul>	<p>I. 5. b II. 5. II. 7. III. 1. III. 2. III. 4.</p>
Sortowanie tablicy metodą bąbelkową, metodą przez wstawianie i wybór – implementacja programów. Porównanie złożoności obliczeniowej poszczególnych typów sortowań. Przeszukiwanie binarne.	<ul style="list-style-type: none"> <li>– sortuje tablicę</li> <li>– potrafi wybrać metodę sortowania w zależności od typu i wielkości tablicy</li> <li>– umie samodzielnie określić złożoność obliczeniową wymienionych algorytmów sortujących</li> </ul>	<p>I. 5. c I. 5. d II. 5. II. 7. II. 8. III. 1. III. 2. III. 3. III. 4.</p>
Obsługa tablicy dwuwymiarowej (macierzy): dodawanie, mnożenie, wyszukiwanie w macierzy kolumny, której suma elementów jest maksymalna, wypełnianie macierzy w charakterystyczny sposób (np. kolejnymi wyrazami zadanego ciągu arytmetycznego).	<ul style="list-style-type: none"> <li>– operuje tablicami dwuwymiarowymi</li> <li>– umie podać przykłady, w których wykorzystuje się tablice dwuwymiarowe, oraz implementuje przykładowe programy (np. gry w kółko i krzyżyk, w statki)</li> <li>– zauważa analogię pomiędzy grami planszowymi a tablicami dwuwymiarowymi</li> </ul>	<p>I. 5. h II. 5. II. 7. II. 8. III. 1. III. 2. III. 3. III. 4.</p>
<b>Srowadzenie macierzy do postaci trójkątnej, rozwiązywanie układu <math>n</math> równań liniowych o <math>n</math> niewiadomych metodą Gaussa – implementacja programu<sup>1</sup>.</b>	<ul style="list-style-type: none"> <li>– potrafi wykorzystać tablice dwuwymiarowe do rozwiązywania zaawansowanych zagadnień numerycznych</li> </ul>	<p>I. 5. g I. 5. h II. 7. II. 8.</p>

<sup>1</sup> Zakładamy, że jest to temat realizowany przez uczniów najzdolniejszych, na przykład w czasie, gdy pozostali piszą prostszy program z wykorzystaniem tablic dwuwymiarowych.

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
	<ul style="list-style-type: none"> <li>– zna podstawowe metody numeryczne stosowane na tablicach</li> <li>– zna niektóre pojęcia związane z teorią macierzy (macierz trójkątna, wyznacznik macierzy)</li> </ul>	III. 1. III. 2. III. 3. III. 4.
<p>Rekurencja – pojęcie, przykłady implementacji programów rekurencyjnych: działanie silnia, ciąg Fibonacciego, potęga o wykładniku naturalnym. Znajdowanie wartości wyrazów ciągów zdefiniowanych rekurencyjnie. Odwracanie ciągu znaków wprowadzonego z klawiatury. Rekurencyjna metoda zamiany liczb pomiędzy systemami liczenia. Schemat Hornera. Złożoność obliczeniowa algorytmów rekurencyjnych.</p>	<ul style="list-style-type: none"> <li>– umie stosować rozwiązania rekurencyjne do grupy zagadnień algorytmicznych</li> <li>– rozumie pojęcie rekurencji</li> <li>– potrafi samodzielnie stworzyć przykłady zastosowań rekurencyjnych i powiązać sytuacje życiowe z rozwiązaniami rekurencyjnymi</li> <li>– poznaje grupę metod sortowania ciągu opartą na rekurencji</li> <li>– potrafi wymienić zarówno zalety, jak i wady rozwiązań rekurencyjnych</li> </ul>	I. 1. I. 5. e I. 5. f II. 5. II. 7. II. 8. III. 1. III. 3. III. 4.
Sortowanie przez scalanie. Sortowanie szybkie (ang. <i>quicksort</i> ).		
<b>Implementacja programu „Wieże Hanoi” w sposób rekurencyjny.</b>	<ul style="list-style-type: none"> <li>– potrafi przeanalizować, określić i rozwiązać problem w sposób rekurencyjny</li> </ul>	II. 5. II. 7. II. 8.
<b>Implementacja programu „Problem skoczka szachowego” lub „Problem ośmiu hetmanów” w sposób rekurencyjny<sup>2</sup>.</b>	<ul style="list-style-type: none"> <li>– zauważa modyfikację czystej metody rekurencyjnej: rekurencję z powrotami</li> <li>– umie zastosować ją w praktyce</li> </ul>	III. 1. III. 2. III. 3. III. 4. III. 7.
Typy strukturalne. Definiowanie własnej, wielopolowej struktury. Przekazywanie zmiennych strukturalnych do funkcji.	<ul style="list-style-type: none"> <li>– potrafi podać przykłady struktur i zastosować je w praktyce</li> <li>– umie utworzyć tablicę przechowującą elementy typu strukturalnego</li> </ul>	II. 8. III. 1. III. 2. III. 5.
Obsługa wejścia/wyjścia. Odczyt z plików, zapis do plików.	<ul style="list-style-type: none"> <li>– umie obsługiwać pliki</li> <li>– potrafi importować dane z plików zewnętrznych oraz eksportować wyniki do plików zewnętrznych</li> </ul>	II. 2. III. 5.
Tworzenie prostej bazy danych z uwzględnieniem możliwości zapisu elementów bazy do pliku i ich odczytu z pliku, sortowania elementów bazy według wybranego klucza oraz wyświetlania elementów bazy.	<ul style="list-style-type: none"> <li>– potrafi zastosować zdobytą wiedzę i umiejętności z zakresu obsługi tablic oraz wykorzystania typów strukturalnych do stworzenia prostej bazy danych</li> <li>– umie utworzyć funkcje obsługujące bazę</li> </ul>	II. 2. II. 8. III. 1. III. 2. III. 5.
Metody szyfrowania: XOR, szyfr Cezara – implementacja programów szyfrujących.	<ul style="list-style-type: none"> <li>– potrafi wymienić kilka metod szyfrowania i deszyfrowania</li> <li>– umie zaimplementować program szyfrujący</li> <li>– proponuje własną metodę szyfrowania</li> </ul>	II. 5. II. 8. III. 1. III. 3. III. 4.
<b>Pojęcie wskaźnika, wskaźniki na zmienne typów prostych, obsługa i przeglądanie tablicy za pomocą wskaźnika, zmiana wartości elementów tablicy. Tablice dynamiczne.</b>	<ul style="list-style-type: none"> <li>– rozpoznaje sytuacje, w których potrzebne jest tworzenie dynamicznych struktur danych</li> <li>– świadomie wybiera metody „pamięciooszczędne”</li> </ul>	I. 8. III. 1. III. 2. III. 3. III. 4.

<sup>2</sup> Propozycja dla uczniów zdolniejszych do realizacji w czasie, gdy inni uczniowie rozwiązują prostsze zadania z zastosowaniem rekurencji.

Zmienne dynamiczne na przykładzie listy jedno- i dwukierunkowej.	<ul style="list-style-type: none"> <li>- stosuje tablice dynamiczne</li> <li>- rozpoznaje sytuacje, w których najkorzystniej jest zastosować strukturę listy</li> <li>- tworzy strukturę listy i ją obsługuje</li> <li>- podaje elementy listy, kasuje wybrane elementy, sortuje elementy listy</li> <li>- potrafi zaprojektować prostą klasę wraz z poprawnymi metodami (np. klasę liczb ułamkowych)</li> <li>- stosuje konstruktory klasy</li> <li>- tworzy obiekty zaprojektowanej klasy</li> </ul>	III. 6. III. 7.
Programowanie obiektowe; pojęcie klasy, obiektu, implementacja programu prezentującego klasę liczb ułamkowych. Pojęcie konstruktora, zastosowanie w projekcie klasy.		
Praca przy grupowym projekcie programistycznym.	<ul style="list-style-type: none"> <li>- potrafi dzielić duży projekt na podproblemy</li> <li>- wykonuje odpowiedzialnie przydzieloną mu część pracy</li> <li>- pozostaje w bieżącym kontakcie z pozostałymi członkami grupy, wykonującymi inną część projektu</li> </ul>	II. 1. II. 2. II. 8. III. 1.

## Bazodanowe funkcje arkusza kalkulacyjnego

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
Wymagania dotyczące tworzenia tabeli nadającej się do wykorzystania mechanizmów bazodanowych arkusza.	<ul style="list-style-type: none"> <li>- zna pojęcie bazy danych</li> <li>- potrafi wymienić, jakie wymagania musi spełniać tabela, która będzie pełnić funkcję bazy danych</li> </ul>	II. 1. II. 8. III. 1.
Autofiltr jako najprostsze narzędzie ułatwiające selekcjonowanie informacji.	<ul style="list-style-type: none"> <li>- potrafi włączyć autofiltr</li> <li>- wykorzystuje autofiltr do selekcji danych</li> </ul>	II. 1. II. 8.
Filtr zaawansowany – narzędzie do wyszukiwania informacji o złożonym kryterium wyboru.	<ul style="list-style-type: none"> <li>- zna zasadę tworzenia filtra zaawansowanego</li> <li>- potrafi wskazać sytuacje wymagające stosowania filtra zaawansowanego</li> <li>- umie wyselekcjonować informacje z tabeli, spełniające złożone kryterium filtrowania</li> </ul>	II. 1. II. 8. III. 4.
Synteza informacji zawartych w tabeli za pomocą sum pośrednich. Użycie funkcji Suma, Licznik, Średnia.	<ul style="list-style-type: none"> <li>- zna pojęcie suma pośrednia i wie, do czego służy to narzędzie</li> <li>- używa sum pośrednich do analizy tabel z zastosowaniem funkcji Suma, Licznik i Średnia</li> </ul>	I. 1. II. 8. III. 4. III. 6.
Tabele przestawne jako narzędzie do automatycznego gromadzenia zbiorczych podsumowań.	<ul style="list-style-type: none"> <li>- wie, do czego służą tabele przestawne i potrafi je zastosować w praktyce</li> </ul>	I. 1. II. 1. II. 8.
Graficzna prezentacja wyników, typy wykresów.	<ul style="list-style-type: none"> <li>- zna możliwości tworzenia różnych typów wykresów</li> <li>- rozumie celowość graficznej interpretacji danych</li> </ul>	II. 1. II. 8. III. 4.
Tworzenie wykresu przestawnego na podstawie tabeli przestawnej.	<ul style="list-style-type: none"> <li>- zna zasady tworzenia wykresu przestawnego i potrafi stworzyć taki wykres</li> </ul>	II. 1. II. 8. III. 4. III. 6.

## Przetwarzanie informacji w relacyjnych bazach danych

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
Pojęcie i przykłady zastosowań relacyjnych baz danych – przypomnienie wiadomości. Prezentacja przykładowej relacyjnej bazy danych.	<ul style="list-style-type: none"> <li>– potrafi uzasadnić konieczność gromadzenia informacji w relacyjnych bazach danych</li> <li>– umie wskazać przykłady baz danych w różnych dziedzinach życia</li> <li>– wyjaśnia zalety elektronicznych baz danych w porównaniu z klasycznymi archiwami gromadzonymi na papierze</li> </ul>	<ul style="list-style-type: none"> <li>I. 1.</li> <li>I. 6.</li> <li>II. 1.</li> <li>II. 3.</li> <li>II. 6.</li> <li>II. 8.</li> <li>III. 1.</li> <li>III. 5.</li> </ul>
Pojęcie relacji. Typy relacji: jeden do jednego, jeden do wielu, wiele do wielu.	<ul style="list-style-type: none"> <li>– potrafi na przykładzie prostej bazy danych zaproponować podział informacji na tabele w taki sposób, by uniknąć powtarzania informacji w bazie</li> <li>– umie samodzielnie połączyć tabele odpowiednimi relacjami</li> </ul>	<ul style="list-style-type: none"> <li>I. 1.</li> <li>II. 1.</li> <li>II. 4.</li> </ul>
Projekt i wykonanie relacyjnej bazy danych dotyczących np. wyników uczniów w zawodach sportowych. Odpowiedni dobór tabel i relacji.	<ul style="list-style-type: none"> <li>– wykorzystuje zdobytą wiedzę do tworzenia baz danych użytecznych w życiu klasy i szkoły</li> <li>– współpracuje z innymi uczniami w celu zgromadzenia danych do wypełnienia rekordami stworzonej bazy</li> </ul>	<ul style="list-style-type: none"> <li>II. 1.</li> <li>II. 2.</li> <li>II. 4.</li> <li>II. 8.</li> <li>III. 1.</li> <li>III. 4.</li> <li>III. 5.</li> <li>III. 7.</li> </ul>
Tworzenie formularza wprowadzania danych za pomocą kreatora.	<ul style="list-style-type: none"> <li>– dba o stworzenie możliwie przyjaznego interfejsu do wprowadzania danych</li> </ul>	<ul style="list-style-type: none"> <li>II. 1.</li> <li>II. 2.</li> <li>II. 8.</li> <li>III. 1.</li> <li>III. 4.</li> <li>III. 5.</li> </ul>
Uzupełnianie stworzonej relacyjnej bazy danych nowymi rekordami, wyszukiwanie rekordów, zastępowanie rekordów.	<ul style="list-style-type: none"> <li>– korzysta z metod wyszukiwania rekordów według prostego klucza</li> <li>– potrafi zmodyfikować rekord, usunąć go lub dodać inny</li> </ul>	<ul style="list-style-type: none"> <li>II. 1.</li> <li>II. 8.</li> <li>III. 5.</li> </ul>
Importowanie danych do tabel z plików tekstowych i z arkusza kalkulacyjnego.	<ul style="list-style-type: none"> <li>– potrafi wykorzystać zapisane informacje, na przykład w edytorze tekstu lub w tabeli arkusza kalkulacyjnego do umieszczenia ich w swojej bazie danych</li> <li>– wyjaśnia zasadność umieszczania informacji w bazie danych</li> </ul>	<ul style="list-style-type: none"> <li>II. 1.</li> <li>II. 2.</li> <li>II. 3.</li> <li>III. 4.</li> <li>III. 5.</li> </ul>
Praca z kwerendami: wyszukującą bezparametryczną i parametryczną oraz aktualizującą.	<ul style="list-style-type: none"> <li>– rozumie pojęcie kwerendy i potrafi wskazać sytuacje wymagające zastosowania kwerend</li> <li>– wskazuje funkcjonalne różnice pomiędzy poszczególnymi rodzajami kwerend</li> </ul>	<ul style="list-style-type: none"> <li>I. 1.</li> <li>II. 4.</li> <li>III. 1.</li> </ul>
Tworzenie kwerend wyszukujących informacje.	<ul style="list-style-type: none"> <li>– potrafi wykorzystać umiejętność pisania kwerendy do wyszukiwania informacji z bazy danych spełniających zadane kryterium</li> </ul>	<ul style="list-style-type: none"> <li>II. 3.</li> <li>II. 4.</li> <li>III. 1.</li> <li>III. 4.</li> </ul>

Zastosowanie kwerendy aktualizującej jako narzędzia do zautomatyzowanego modyfikowania bazy danych.	– tworzy użyteczne kwerendy usprawniające aktualizację bazy danych	I. 1. II. 4. III. 1. III. 5.
Generowanie raportów z bazy danych za pomocą kreatorów.	– umie wygenerować z bazy danych raport zawierający wybrane informacje	I. 1. II. 1. II. 2. II. 3. II. 8.
Tworzenie i korzystanie z formularzy, dopisywanie i modyfikacja danych z wykorzystaniem formularzy.	– dba o estetyczną stronę bazy danych – poznaje mechanizmy ułatwiające operowanie na bazie danych – tworzy formularze ułatwiające praktyczne zastosowanie stworzonej bazy danych	II. 1. II. 4.
Mechanizmy ochrony bazy danych.	– wskazuje zagrożenia spowodowane brakiem zabezpieczeń bazy danych – potrafi stosować mechanizmy ochrony danych przed nieupoważnioną bądź przypadkową modyfikacją	I. 1. II. 1. III. 4.
Podstawy języka SQL, zasady składni i podstawowe instrukcje.	– wie, co to jest język zapytań SQL i do czego służy – zna podstawowe polecenia języka i zasady składni	II. 4. III. 2. III. 5.
Analiza poleceń zapisanych w języku SQL.	– potrafi analizować fragmenty kodów napisanych w języku SQL, wyjaśniać polecenia i użyte komendy	II. 4. III. 5. III. 6.

## Systemy i sieci komputerowe

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
Obsługa i konserwacja używanego systemu operacyjnego.	– opisuje budowę nazwy pliku oraz jego atrybuty w systemie Windows – poprawnie definiuje pojęcia: wirus komputerowy, robak, trojan oraz wyjaśnia ich działanie – wyjaśnia pojęcie kompresji danych – wyszukuje pliki według zadanych kryteriów (np. data, nazwa, rozszerzenie itp.) i nadaje plikom atrybuty – uruchamia i konfiguruje program antywirusowy – tworzy kopie zapasowe systemu i danych oraz odzyskuje pliki usunięte za pomocą odpowiednich programów narzędziowych	I. 2. I. 3. II. 1. II. 6. III. 8.
<b>Podstawy obsługi systemu Linux.</b>	– umie wykonać podstawowe operacje systemu Linux: przeglądać zasoby w systemie, kopiować, usuwać pliki, sprawdzać i ustawiać prawa dostępu do swoich zasobów, zmieniać hasło	I. 2. I. 3. II. 1. II. 3.

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
Topologie i technologie sieciowe. Rodzaje sieci ze względu na zasięg i architekturę, media transmisji, urządzenia usprawniające pracę sieci. Pojęcie protokołu i rodzaje protokołów, pojęcie serwera i rodzaje serwerów.	<ul style="list-style-type: none"> <li>– zna podstawowe pojęcia związane z pracą sieci, na przykład: protokół połączeniowy i bezpołączeniowy, serwer DNS, serwer dedykowany, ruting dynamiczny, pakiet danych itp.</li> <li>– zna nazewnictwo i zastosowanie urządzeń pracujących w sieci</li> </ul>	I. 1. I. 2. II. 3. III. 7.
Projektowanie prostych sieci komputerowych.	<ul style="list-style-type: none"> <li>– umie zaprojektować prostą sieć lokalną z uwzględnieniem warunków, w jakich projekt ma być realizowany, oraz funkcji, jakie ma spełniać, z możliwością wyjścia do Internetu</li> <li>– potrafi udostępnić zasoby komputera do sieci</li> </ul>	II. 3. II. 6. II. 8. III. 4. III. 6. III. 7.
Adresacja w sieci Internet, budowa adresu IP.	<ul style="list-style-type: none"> <li>– potrafi zinterpretować adres IP</li> <li>– umie wymienić i wyjaśnić poszczególne elementy adresu, wyjaśnić znaczenie serwera DNS</li> </ul>	I. 2. II. 1. II. 3.
Bezpieczeństwo w sieci.	<ul style="list-style-type: none"> <li>– wymienia zagrożenia płynące z pracy komputera w sieci</li> <li>– zna środki i metody bezpiecznej pracy w sieci</li> <li>– zna pojęcia związane z zabezpieczeniem sieci, na przykład: zapora ogniowa (ang. <i>firewall</i>)</li> <li>– konfiguruje program do zabezpieczenia zasobów komputera</li> <li>– zna podstawy kodowania danych</li> </ul>	II. 6. III. 7. III. 8.

## Multimedia

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
Dźwięk w komputerze.	<ul style="list-style-type: none"> <li>– potrafi wymienić kilka typów plików dźwiękowych</li> <li>– wyjaśnia różnice i obszary zastosowań poszczególnych formatów</li> <li>– wymienia metody kompresji plików dźwiękowych</li> <li>– podaje zasady poszczególnych rodzajów kompresji</li> </ul>	I. 1. I. 2. III. 7.
Grafika 2D.	<ul style="list-style-type: none"> <li>– wymienia formaty plików bitmapowych</li> <li>– zna podstawowe pojęcia związane z grafiką bitmapową</li> <li>– przedstawia metody kompresji plików graficznych</li> <li>– wyjaśnia podstawowe różnice pomiędzy grafiką wektorową a bitmapową</li> </ul>	I. 1. II. 2. II. 6. II. 8. III. 7.

Podstawy grafiki 3D.	<ul style="list-style-type: none"> <li>- potrafi wyjaśnić zasadę tworzenia sceny 3D metodą „śledzenia promienia”</li> <li>- tworzy prostą scenę 3D za pomocą programu komputerowego</li> </ul>	I. 1. II. 6. II. 8.
Animacja komputerowa.	<ul style="list-style-type: none"> <li>- zna zasady łączenia obrazów statycznych w animację</li> <li>- tworzy własną animację w postaci połączonych kilku klatek, np. w formacie gif</li> </ul>	I. 1. II. 6. II. 8. III. 7.
Tworzenie stron WWW.	<ul style="list-style-type: none"> <li>- zna podstawowe znaczniki HTML i potrafi za ich pomocą stworzyć statyczną stronę internetową z wykorzystaniem stylu, zawierającą tabele, hiperłącza, grafikę, dźwięk</li> <li>- wyszukuje w Internecie potrzebne informacje i wykorzystuje je do tworzenia własnych stron</li> <li>- zna i stosuje obowiązujące zasady prawne dotyczące prawa autorskiego</li> <li>- umieszcza w kodzie strony istniejące skrypty pisane w Javie</li> </ul>	I. 4. II. 1. II. 2. II. 3. II. 6. III. 8.

## Tendencje w rozwoju informatyki i jej zastosowań

Treści nauczania	Cele edukacyjne i umiejętności ucznia	Standardy wymagań
Historia i stan obecny architektury komputerów i systemów operacyjnych.	<ul style="list-style-type: none"> <li>- potrafi odnaleźć i zaprezentować informacje dotyczące zmian w budowie komputerów i systemów operacyjnych</li> <li>- dyskutuje na temat możliwych scenariuszy rozwoju</li> <li>- wyszukuje informacje na temat stosowania urządzeń komputerowych w różnych dziedzinach życia</li> </ul>	I. 1. I. 2. I. 3. I. 6. II. 1. II. 3. III. 7. III. 8.
Przedstawienie z najnowszych osiągnięć technik informacyjnych i informatyki.	<ul style="list-style-type: none"> <li>- zna najnowsze osiągnięcia w rozwoju technik informacyjnych</li> <li>- śledzi na bieżąco nowe trendy z nimi związane</li> <li>- potrafi określić ważniejsze etapy w rozwoju informatyki dotyczące pojęć, metod i środków technicznych</li> </ul>	I. 1. I. 2. I. 3. I. 6. II. 1. II. 3. III. 7. III. 8.
Informacje o możliwościach komputerów i ich granicach.	<ul style="list-style-type: none"> <li>- potrafi wymienić możliwości nowoczesnych komputerów</li> <li>- ocenia możliwości i granice stosowania komputerów (systemy równoległe, sieci neuronowe – sztuczna inteligencja)</li> </ul>	I. 1. I. 3. I. 6. II. 1. II. 3. III. 7. III. 8.

## VIII. Proponowany rozkład materiału

Proponowany przez nas rozkład materiału przeznaczony jest dla liceum ogólnokształcącego, dla klas realizujących informatykę w zakresie rozszerzonym, przy założeniu, że w klasie pierwszej zaliczony został przedmiot technologia informacyjna (w zakresie podstawowym). Przyjęliśmy pięć godzin lekcyjnych w cyklu dwóch lat kształcenia, przy czym w drugiej klasie są to dwa semestry po 3 godziny lekcyjne tygodniowo<sup>3</sup>, co daje około 90 godzin, a w trzeciej klasie przewidujemy dwa semestry po dwie godziny lekcyjne tygodniowo, czyli 50 godzin.

### Klasa druga

Dział	Tematy	Liczba godzin
Algorytmika	Lekcja organizacyjna, przedstawienie programu nauczania i regulaminu pracowni informatycznej. Zapoznanie ucznia z kryteriami oceny, sprawdzania wiadomości i wymaganiami edukacyjnymi.	1
	Reprezentacja danych w komputerze – systemy liczbowe. Przypomnienie wiadomości o logicznej budowie komputera.	2
	Pojęcie algorytmu, przykłady, wyszczególnienie etapów algorytmu.	1
	Specyfikacja problemu i poprawność algorytmu; cechy poprawnego algorytmu.	1
	Sposoby zapisu algorytmu (pseudojęzyk, schemat blokowy, lista kroków), zastosowanie w przykładach dla zadanych problemów.	1
	Ćwiczenia w tworzeniu i odczytywaniu algorytmów na podstawie schematów blokowych.	2
	Analiza porównawcza algorytmów – podstawy, wprowadzenie do zagadnienia złożoności obliczeniowej.	2
	Sprawdzian wiadomości z systemów liczbowych oraz zapisu algorytmów i ich analizy.	1
	Omówienie i poprawa sprawdzianu.	1
Programowanie	Charakterystyka wybranego języka programowania (C++) i omówienie środowiska programistycznego (Borland Builder 6).	1
	Podstawowe typy zmiennych i operacje standardowego wejścia/wyjścia – implementacja pierwszego programu.	2
	Instrukcje: przypisania i porównania; podstawowe operatory matematyczne i logiczne – zastosowanie w implementacji programu.	2
	Instrukcja warunkowa i instrukcja wyboru: implementacja programu rozwiązującego równanie liniowe z jedną niewiadomą.	2

<sup>3</sup> Liczba godzin przeznaczonych na realizację tematów wynosi w rozkładzie 90, pozostałe godziny przeznaczone są na klasyfikacje semestralne i na godziny do dyspozycji nauczyciela (realizacja i implementacja programów wybranych przez nauczyciela w celu ugruntowania wiedzy uczniów, dodatkowe ćwiczenia, realizacja proponowanych w programie dodatkowych tematów, które umieszczone są w tabeli: treści rozszerzające umiejętności ucznia).

Rozwiązywanie równania kwadratowego – implementacja programu – praca w grupach.	1
Instrukcje pętli; implementacja programu: „Zgadnij, jaką liczbę wybrał komputer” <sup>4</sup> .	2
Omówienie i implementacja algorytmów sprawdzających własności liczb całkowitych (badanie podzielności liczb, sprawdzanie, czy liczba jest pierwsza).	2
Algorytm Euklidesa – omówienie i implementacja programu.	1
Zapoznanie z pojęciem metody Monte Carlo i implementacja programu: znajdowanie przybliżonej wartości liczby $\pi$ lub ruchu Browna – błędzenie przypadkowe. Wspólna analiza otrzymanych wyników.	2
Przybliżanie wartości pierwiastka kwadratowego – przedstawienie metody i implementacja programu (metoda Newtona–Raphsona).	2
Omówienie metody wyznaczania przybliżonej wartości miejsca zerowego funkcji przez połowienie przedziałów (bisekcję); implementacja programu.	2
Praktyczny sprawdzian dotyczący umiejętności z zakresu poznanego przez ucznia.	2
Omówienie i poprawa sprawdzianu.	1
Funkcje w C++; sposób przekazywania parametrów do funkcji: parametry formalne i aktualne; zmienne globalne i lokalne; prototypy funkcji.	2
Tablica jednowymiarowa; deklaracja, inicjalizacja, przekazywanie tablic do funkcji; implementacja prostego programu wykorzystującego zastosowanie tablicy jednowymiarowej.	2
Implementacja programu zamieniającego liczbę z systemu dziesiętnego na inny system liczbowy.	2
Przeszukiwanie tablicy jednowymiarowej; znajdowanie elementu maksymalnego (minimalnego) tablicy.	1
Przeszukiwanie tablicy w celu znalezienia wyróżnionego elementu – wersja z wartownikiem i bez wartownika, porównanie obu metod pod kątem optymalności algorytmu.	2
Sito Eratostenesa – implementacja programu.	1
Sortowanie tablicy jednowymiarowej metodą bąbelkową. Omówienie złożoności obliczeniowej metody.	2
Sortowanie tablicy metodami przez wybór i selekcję.	2
Omówienie metody „dziel i zwyciężaj”; przeszukiwanie binarne.	2
Tablica dwuwymiarowa; wypełnianie tablicy oraz znajdowanie sumy elementów leżących na przekątnej macierzy kwadratowej.	2
Implementacja przykładowego programu z wykorzystaniem operacji na tablicach dwuwymiarowych.	2

<sup>4</sup> Użytkownik odgaduje wartość liczby całkowitej wygenerowanej losowo przez program z danego zakresu – po każdym wskazaniu liczby otrzymuje informację, czy jest ona zbyt duża, czy zbyt mała.

Dział	Tematy	Liczba godzin
	Praktyczny sprawdzian dotyczący umiejętności implementacji programów z zastosowaniem tablic jedno- i dwuwymiarowych i zdefiniowanych przez użytkownika funkcji, w tym funkcji sortujących.	2
	Omówienie i poprawa sprawdzianu.	1
	Pojęcie rekurencji, przykłady; implementacja programów obliczających silnię z liczby, potęgę, kolejne wyrazy ciągu Fibonacciego.	2
	Schemat Hornera – implementacja programu, porównanie metody z obliczaniem wartości wielomianu przy wykorzystaniu postaci definicyjnej wielomianu.	2
	Ćwiczenia w implementacji programów o rozwiązaniu rekurencyjnym <sup>5</sup> .	2
	Sortowanie przez scalanie.	2
	Sortowanie szybkie (ang. <i>quicksort</i> ).	2
	Porównanie metod iteracyjnej i rekurencyjnej dla przykładowego zadania algorytmicznego (złożoność obliczeniowa czasowa i pamięciowa).	2
	Pojęcie struktury. Definiowanie własnej, wielopolowej struktury, obsługa danych typu strukturalnego poprzez funkcje.	2
	Implementacja programu z zastosowaniem tablicy o elementach typu strukturalnego: prosta baza danych <sup>6</sup> . Obsługa bazy (wypełnienie bazy, sortowanie wg zadanego klucza; stabilność sortowania).	2
	Obsługa wejścia/wyjścia. Odczyt z plików, zapis do plików. Obsługa plików tekstowych i plików binarnych.	4
	Metody szyfrowania, omówienie i implementacja wybranych metod (szyfr Cezara, XOR).	2
	Praktyczny sprawdzian umiejętności implementacji programów z zastosowaniem całości wiedzy zdobytej w klasie drugiej.	2
	Poprawa i omówienie sprawdzianu.	1
	Przeszukiwanie tekstu w poszukiwaniu wzorca – metoda naiwna.	2
	Omówienie i implementacja programu „Odwrotna notacja polska” dla wyrażeń o określonej długości.	2
	Praca z przykładowym arkuszem maturalnym z zakresu „Algorytmika i programowanie”.	2
	Podsumowanie wiedzy i umiejętności zdobytych w klasie drugiej, implementacja przykładowych programów.	2
	Klasyfikacja końcoworoczna.	1

<sup>5</sup> Uczniowie zdolniejsi, po zasygnalizowaniu przez nauczyciela zagadnienia na lekcji poprzedniej, implementują rozwiązania problemu ośmiu hetmanów i skoczka szachowego. Inni mogą w tym czasie rozwiązywać na przykład problem wież Hanoi.

<sup>6</sup> Uczniowie indywidualnie wybierają elementy swojej bazy: może to być komis samochodowy, baza uczniów, biblioteka itp. Ze względu na ograniczone umiejętności ucznia na tym etapie nauczania zakładamy, że baza jest implementowana na statycznej tablicy, na przykład dziesięcioelementowej.

Propozycje tematów do zrealizowania na godzinach przeznaczonych do dyspozycji nauczyciela (18 godzin)		
	Pojęcie wskaźnika, przykłady zastosowań wskaźników na typy proste.	3
	Tablice dynamiczne – alokacja i usuwanie z pamięci, wypełnianie i przeglądanie tablicy jednowymiarowej za pomocą wskaźnika.	2
	Charakterystyka struktur dynamicznych na przykładzie listy jedno- i dwukierunkowej.	4
	Wyznaczanie metodą siecznych przybliżonego miejsca zerowego wielomianu – implementacja programu.	2
	Obliczanie całki Riemanna z definicji i metodą Monte Carlo – omówienie zagadnienia i implementacja jednej z metod.	2
	Wstęp do programowania obiektowego; pojęcie klasy, konstrukcja klasy, przykłady <sup>7</sup> .	3
	Klasa liczb ułamkowych (lub zespolonych) – obiektowa implementacja programu „kalkulator liczb ułamkowych (lub zespolonych)”.	2

### Klasa trzecia

Dział	Tematy	Liczba godzin
Przetwarzanie informacji w relacyjnych bazach danych	Relacyjna baza danych i podstawowe pojęcia z nią związane – przypomnienie wiadomości i definicji.	1
	Typy relacji: jeden do jednego, jeden do wielu, wiele do wielu; praktyczne wykorzystanie umiejętności tworzenia relacji w gotowej bazie danych.	1
	Typy kwerend, pisanie kwerend wyszukiwujących i aktualizujących (kwerendy parametryczne).	2
	Tworzenie raportów.	1
	Mechanizmy ochrony bazy danych.	1
	Podstawy języka SQL, zasady składni i podstawowe instrukcje.	2
	Sprawdzian wiadomości dotyczący podstawowych pojęć z zakresu baz danych i praktycznych umiejętności tworzenia bazy danych na podstawie danych z pliku tekstowego.	2
	Omówienie i poprawa sprawdzianu.	1

<sup>7</sup> Zakładamy, że treści związane z programowaniem obiektowym będą realizowane tylko w grupie uczniów szczególnie zdolnych i zainteresowanych pogłębianiem wiedzy. W każdym innym przypadku korzystniej jest przeznaczyć te godziny na realizację prostszych algorytmów o niższym stopniu trudności.

<b>Dział</b>	<b>Tematy</b>	<b>Liczba godzin</b>
<b>Systemy i sieci komputerowe</b>	Przypomnienie i rozszerzenie wiadomości o budowie komputera.	1
	System operacyjny i programy narzędziowe – mechanizmy ochrony danych.	1
	Sieci komputerowe – rodzaje sieci ze względu na topologię i zasięg; zalety, wady, urządzenia działające w sieci.	2
	Pojęcia związane z funkcjonowaniem sieci komputerowych (protokół, pakiet, DNS, ruting itp.), adresacja w sieci Internet, adresacja w sieci. Warstwowy model budowy sieci.	2
	Usługi sieciowe, udostępnianie zasobów komputera do sieci.	1
	Bezpieczeństwo w sieci – programy chroniące zasoby komputera.	1
	Sprawdzian wiadomości dotyczących systemów i sieci komputerowych.	1
	Omówienie i poprawa sprawdzianu.	1
<b>Multimedia</b>	Dźwięk w komputerze: podstawowe formaty plików dźwiękowych: midi, wave i mp3 oraz metody kompresji dźwięku.	2
	Grafika bitmapowa (rastrowa) – formaty plików, parametry obrazu rastrowego i metody kompresji plików graficznych.	2
	Grafika wektorowa a grafika rastrowa – podstawowe różnice w tworzeniu i zapisie obrazu.	1
	Zasady tworzenia obrazu 3D i podstawowe pojęcia: rendering, ray tracing. POV-Ray – program do tworzenia grafiki 3D.	1
	Tworzenie własnej sceny 3D w programie POV-Ray z zastosowaniem przesunięć i obrotów.	2
	Tworzenie animacji z wykorzystaniem scen wygenerowanych przez program POV-Ray.	1
	Przypomnienie z gimnazjum znaczników HTML.	1
	Zastosowanie stylów. Osadzanie skryptów Javy w kodzie HTML.	1
	Wykonanie własnej strony na zadany temat na podstawie informacji wyszukanych w sieci Internet.	2
	Sprawdzian wiadomości dotyczących multimediiów.	2
	Omówienie i poprawa sprawdzianu.	1

<b>Tendencje w rozwoju informatyki i jej zastosowań</b>	Perspektywy rozwoju informatyki <sup>8</sup> (proponowane tematy: mobilny dostęp do Internetu, superkomputery, komputery optyczne i DNA, fotografia cyfrowa, komórkowe systemy operacyjne, komputery równoległe, sztuczna inteligencja itd.).	4
<b>Powtórzenie wiadomości</b>	Ćwiczenia indywidualne i grupowe z przykładowymi arkuszami maturalnymi (zgodnymi z formułą nowej matury z podziałem na zadania otwarte i zamknięte).	8
	Klasyfikacja końcoworoczna.	1
<b>Propozycja tematów do zrealizowania na godzinach przeznaczonych do dyspozycji nauczyciela (8 godzin)</b>		
<b>System Linux</b>	System operacyjny Linux – wiadomości ogólne, historia systemu i perspektywy rozwoju. Logowanie się do systemu, wylogowywanie z systemu, poprawne zamykanie.	1
	Budowa systemu plików – podstawowe polecenia operujące na plikach i katalogach (przeglądanie, kopiowanie, usuwanie, zmiana nazwy). Oprogramowanie użytkowe systemu Linux.	1
	Interpretowanie i ustalanie praw dostępu w systemie Linux. Przeglądanie i usuwanie działających procesów.	1
	Wykorzystanie środowiska graficznego systemu Linux (KDE, GNOME). Stosowanie narzędzi TI dostępnych w systemie Linux (pakiet biurowy, edytor grafiki).	1
	Montowanie urządzeń. Wymiana plików pomiędzy systemami operacyjnymi.	1
<b>Bazodanowe funkcje arkusza kalkulacyjnego</b>	Importowanie danych z plików tekstowych. Autofiltr i sortowanie. Filtr zaawansowany – złożone kryteria wyszukiwania – ćwiczenia.	1
	Sumy pośrednie – suma, średnia, licznik – przypomnienie wiadomości. Ćwiczenia w tworzeniu sum.	1
	Tabele i wykresy przestawne. Mechanizmy ochrony arkusza i komórki.	1

## IX. Kontrola i ocena osiągnięć ucznia

### Klasyfikacja wymagań

Ocena pełni rolę zarówno informującą, jak i motywującą do pracy, dlatego należy zapoznać uczniów z kryteriami oceniania i z rozróżnieniem na materiał teoretyczny i umiejętność rozwiązywania zadań problemowych. Uczeń musi mieć świadomość, jakie powinien posiadać umiejętności i wiadomości, aby otrzymać daną ocenę, dlatego na każdej pierwszej godzinie lekcyjnej rozpoczynającej dany dział należy przedstawić wymagania i kryteria oceniania. Zestaw wymagań do poszczególnych treści nauczania powinien być adekwatny do oceny, jaką uczeń otrzymuje za nabytą wiedzę i opanowane umiejętności.

<sup>8</sup> Tematy tego działu programowego mogą być realizowane za pomocą prezentacji wykonywanych przez uczniów. Nauczyciel może wówczas ocenić umiejętności ucznia w zakresie tworzenia czytelnych i poprawnych pokazów slajdów.

Korelacja pomiędzy zestawem wymagań a oceną przedstawia się następująco:

- spełnienie wymagań **koniecznych (K)** odpowiada ocenie **dopuszczającej**;
- spełnienie wymagań **koniecznych i podstawowych (P)** odpowiada ocenie **dostatecznej**;
- spełnienie wymagań **koniecznych, podstawowych i rozszerzających (R)** odpowiada ocenie **dobrej**;
- spełnienie wymagań **koniecznych, podstawowych, rozszerzających i dopełniających (D)** odpowiada ocenie **bardzo dobrej**;
- spełnienie wymagań **koniecznych, podstawowych, rozszerzających, dopełniających i wykraczających (W)** odpowiada ocenie **celującej**.

Wymagania konieczne stanowią bazę do rozszerzania i pogłębiania dalszej wiedzy, nie ma zatem możliwości wystawienia uczniowi oceny pozytywnej, jeśli ich nie spełni. Wymagania na poziomie koniecznym powinny stanowić nie mniej niż 40% wszystkich wymagań. Wymagania podstawowe, rozszerzające i dopełniające stanowią po około 20% każde.

Inaczej traktujemy wymagania wykraczające, które występują w naszym programie z reguły zamiennie z wymaganiami niższych rzędów. Uznajemy bowiem, że stanowią one rozszerzenie wymagań dopełniających dla uczniów zdolniejszych i szczególnie zainteresowanych przedmiotem. Obejmują one treści wykraczające poza program nauczania. Dla uczniów zdolniejszych stanowią zachętę do poszerzania swojej wiedzy i pomagają w przyszłym świadomym wyborze dalszego kierunku kształcenia.

Wychodzimy z założenia, że uczeń, który otrzymuje ocenę dopuszczającą, musi samodzielnie przyswoić wiedzę teoretyczną dotyczącą danego zagadnienia, natomiast niektóre czynności, takie jak implementacja programów w języku programowania, wykonuje poprawnie przy pomocy nauczyciela. Pomoc nauczyciela sprowadza się tu jednak do weryfikowania pomysłów ucznia i naprowadzania go na właściwe rozwiązania; nauczyciel nie pełni funkcji pomocnika, który wykonuje zadanie z uczniem bądź za niego.

### Propozycja oceny według klasyfikacji wymagań

Poniżej przedstawiamy w sposób tabelaryczny naszą propozycję oceniania ucznia zgodnie z programem nauczania uzupełnionym o treści nieobowiązkowe.

Umiejętność nabyta przez ucznia	Wymagania				
	K	D	P	R	W
<b>Algorytmika i programowanie</b>					
zna pozycyjne systemy liczbowe i potrafi przeliczać liczby zapisane w jednym systemie na inny	•	•	•	•	•
zna pojęcie algorytmu, umie podać przykłady zadań algorytmicznych	•	•	•	•	•
potrafi zapisać prosty algorytm w pseudojęzyku lub za pomocą listy kroków	•	•	•	•	•
zapisuje prosty algorytm za pomocą schematu blokowego		•	•	•	•
potrafi wymienić cechy algorytmu		•	•	•	•

umie wyjaśnić, co to jest algorytm liniowy	•	•	•	•	•
wyjaśnia, co to jest algorytm rozgałęziony	•	•	•	•	•
wyodrębnia części składowe algorytmu	•	•	•	•	•
omawia klasyczne, proste algorytmy		•	•	•	•
umie dokonać analizy algorytmu			•	•	•
potrafi określić złożoność obliczeniową algorytmu				•	•
umie obsługiwać narzędzie służące do implementacji programów w wybranym języku programowania	•	•	•	•	•
pisze kod źródłowy w sposób przejrzysty, stosuje komentarze i wcięcia	•	•	•	•	•
zna podstawowe elementy języka programowania	•	•	•	•	•
rozumie pojęcia: stałe, zmienne i umie podać przykłady ich zastosowania	•	•	•	•	•
potrafi zadeklarować zmienne prostych typów w pisanim kodzie	•	•	•	•	•
umie zaimplementować, skompilować i uruchomić prosty program liniowy	•	•	•	•	•
rozumie pojęcia: funkcja, procedura i umie je wykorzystać przy implementacji programu			•	•	•
zna operatory matematyczne i umie je zastosować w programie	•	•	•	•	•
zna operatory logiczne i relacji, potrafi je zastosować w programie		•	•	•	•
umie zastosować typ tablicowy w implementacji programów (tablica jednowymiarowa o elementach typu prostego)		•	•	•	•
rozumie pojęcie struktury i potrafi podać przykłady zmiennych strukturalnych	•	•	•	•	•
definiuje strukturę składającą się z kilku pól		•	•	•	•
umie zadeklarować tablicę jednowymiarową o elementach typu zdefiniowanej przez siebie struktury			•	•	•
potrafi przekazać do funkcji jako parametr formalny tablicę jednowymiarową o elementach typu prostego			•	•	•
umie przekazać do funkcji jako parametr formalny tablicę jednowymiarową o elementach typu strukturalnego				•	•
stosuje typ tablicowy w implementacji programów (tablica mająca dwa wymiary lub więcej)				•	•
potrafi zaimplementować program przeszukujący ciąg znaków w celu odnalezienia wyróżnionego elementu	•	•	•	•	•
potrafi zaimplementować program porządkujący ciąg elementów metodą bąbelkową		•	•	•	•

Umiejętność nabyta przez ucznia	Wymagania				
	K	D	P	R	W
potrafi zaimplementować program porządkujący ciąg elementów metodą sortowania przez wstawianie			•	•	•
potrafi zaimplementować program porządkujący ciąg elementów metodą sortowania przez selekcję			•	•	•
potrafi zaimplementować iteracyjnie programy realizujące proste metody numeryczne i metodę Monte Carlo			•	•	•
zna pojęcie rekurencji, umie podać proste przykłady jej zastosowania (ciąg Fibonacciego, silnia, potęga)	•	•	•	•	•
umie samodzielnie zaimplementować proste programy rekurencyjne (jak powyżej)			•	•	•
potrafi napisać program rozwiązujący klasyczny problem wież Hanoi				•	•
<b>umie przeanalizować i zaimplementować program klasycznego problemu skoczka szachowego<sup>9</sup></b>					•
<b>potrafi przeanalizować i rozwiązać implementacyjnie problem ośmiu hetmanów</b>					•
umie wyjaśnić, na czym polega metoda sortowania szybkiego			•	•	•
potrafi samodzielnie zaimplementować program sortujący metodą szybką				•	•
zna schemat Hornera i umie go zaimplementować			•	•	•
zna algorytm Euklidesa i umie go zaimplementować			•	•	•
zna metody szyfrowania, implementuje przynajmniej jedną z nich			•	•	•
implementuje program, na przykład kolejnych przybliżeń miejsca zerowego funkcji wielomianowej			•	•	•
<b>wykorzystuje w programach funkcje zdefiniowane w plikach bibliotecznych napisanych przez siebie</b>				•	•
<b>zna pojęcie wskaźnika, umie zastosować wskaźnik do obsługi zmiennych typów prostych</b>				•	•
<b>stosuje wskaźniki przy pracy z tablicami</b>				•	•
<b>tworzy tablice dynamiczne i obsługuje je za pomocą wskaźnika</b>					•
<b>zna pojęcie klasy i obiektu, umie zaprojektować własną klasę</b>					•
<b>Bazodanowe funkcje arkusza kalkulacyjnego</b>					
<b>importuje dane do tabel z pliku tekstowego</b>			•	•	•

<sup>9</sup> Proponujemy, aby w czasie, gdy uczeń słabszy implementuje prosty program rekurencyjny, uczeń starający się o ocenę wyższą wykonał zadanie trudniejsze.

sortuje tabelę będącą bazą danych	•	•	•	•	•
używa autofiltra do tabel	•	•	•	•	•
potrafi za pomocą filtra zaawansowanego wyselekcjonować dane spełniające złożone kryterium			•	•	•
wie, do czego służą sumy pośrednie		•	•	•	•
umie odpowiednio przygotować tabelę do zastosowania sum pośrednich		•	•	•	•
stosuje sumy pośrednie do syntezy informacji zawartych w tabelach			•	•	•
wie, do czego służą tabele przestawne		•	•	•	•
generuje tabelę przestawną i poprawnie interpretuje jej wyniki			•	•	•
umie wygenerować wykres przestawny i potrafi go właściwie zinterpretować				•	•
wskazuje przykłady, kiedy warto zastosować graficzną prezentację danych	•	•	•	•	•
potrafi wskazać i stworzyć najlepszy typ wykresu dla określonego typu zadania		•	•	•	•
poprawnie interpretuje wykres i odpowiednio modyfikuje jego ustawienia: skala, nazwa osi, legenda, kolory, linie, tło itp.			•	•	•
<b>Przetwarzanie informacji w relacyjnych bazach danych</b>					
potrafi wskazać przykłady baz danych	•	•	•	•	•
umie wymienić charakterystyczne elementy relacyjnej bazy danych	•	•	•	•	•
rozumie zasadność gromadzenia informacji w wielu tabelach	•	•	•	•	•
projektuje tabele składowe relacyjnej bazy danych	•	•	•	•	•
umie sformatować kolumny tabeli odpowiednio do typu danych	•	•	•	•	•
modyfikuje dane w tabeli: kopiuje, dodaje, usuwa rekordy	•	•	•	•	•
potrafi tworzyć właściwe relacje pomiędzy tabelami	•	•	•	•	•
wymusza i testuje więzy integralności			•	•	•
projektuje prosty formularz kolumnowy	•	•	•	•	•
potrafi zaprojektować formularz z podformularzami			•	•	•
porządkuje tabelę i przegląda wybrane rekordy		•	•	•	•
zna pojęcie kwerendy i potrafi wymienić jej typy	•	•	•	•	•
projektuje proste kwerendy wybierające i potrafi je modyfikować	•	•	•	•	•

Umiejętność nabyta przez ucznia	Wymagania				
	K	D	P	R	W
definiuje złożone kryteria wyboru rekordów		•	•	•	•
projektuje kwerendy aktualizujące				•	•
definiuje i formatuje pola obliczeniowe			•	•	•
projektuje zapytania parametryczne			•	•	•
potrafi tworzyć raporty za pomocą kreatorów		•	•	•	•
umie tworzyć raporty szczegółowe z zastosowaniem obliczeń			•	•	•
eksportuje dane z bazy danych do edytora tekstu, arkusza kalkulacyjnego		•	•	•	•
importuje dane z pliku tekstowego i arkusza kalkulacyjnego		•	•	•	•
potrafi stworzyć wykres ilustrujący zależności pomiędzy danymi w bazie danych				•	•
stosuje mechanizmy ochrony bazy danych				•	•
zna składnię i podstawowe komendy języka SQL					•
<b>Systemy i sieci komputerowe</b>					
biegle wykonuje podstawowe operacje na plikach w systemie Windows (wyszukiwanie, kopiowanie, kasowanie, zmiana nazwy i atrybutów)	•	•	•	•	•
zna mechanizmy zabezpieczania danych		•	•	•	•
zna i poprawnie interpretuje pojęcia: wirus komputerowy, robak, trojan			•	•	•
poprawnie instaluje i konfiguruje program antywirusowy				•	•
zna podstawy pracy w systemie operacyjnym Linux (loguje się do systemu, sprawdza zawartość swojego katalogu, wylogowuje się i poprawnie zamyka system)			•	•	•
zna i wykorzystuje podstawowe polecenia systemu Linux (kopiuje, kasuje pliki, zmienia swoje hasło, sprawdza, kto aktualnie jest zalogowany w systemie itp.)				•	•
interpretuje prawa dostępu użytkowników		•	•	•	•
potrafi ustalić prawa dostępu użytkowników do swoich zasobów			•	•	•
montuje urządzenia w systemie				•	•
przegląda działające procesy w systemie Linux, potrafi przerwać wskazany proces				•	•
korzysta z narzędzi TI dostępnych w systemie Linux (pakiet biurowy, edytor grafiki)			•	•	•
wymienia pliki pomiędzy systemami				•	•

klasyfikuje sieci komputerowe ze względu na zasięg i typ połączeń	•	•	•	•	•
zna rodzaje topologii sieci LAN, wymienia je, określa cechy każdej z nich			•	•	•
zna pojęcie protokołu i potrafi wymienić kilka rodzajów protokołów				•	•
wymienia i opisuje urządzenia usprawniające pracę sieci			•	•	•
zna pojęcie serwera, potrafi wymienić kilka typów serwerów		•	•	•	•
umie zaprojektować lokalną sieć komputerową				•	•
potrafi sieciowo udostępnić zasoby komputera			•	•	•
szczegółowo wyjaśnia znaczenia segmentów adresu IP				•	•
<b>umie wymienić poszczególne warstwy modelu sieci ISO/OSI</b>					•
zna sposoby zabezpieczania komputera pracującego w sieci			•	•	•
zna metody szyfrowania danych w sieci komputerowej				•	•
korzysta z przeglądarek i wyszukiwarek internetowych	•	•	•	•	•
selekcjonuje informacje uzyskane w sieci	•	•	•	•	•
korzysta z aplikacji służących do przesyłania w sieci plików			•	•	•
potrafi utworzyć prostą stronę internetową z zachowaniem dbałości o czytelność i estetykę	•	•	•	•	•
umieszcza na zaprojektowanej przez siebie stronie tabelkę, listy punktowane i numerowane, tytuły, grafikę		•	•	•	•
poprawnie formatuje tekst na tworzonej przez siebie stronie WWW w celu wyróżnienia niektórych treści (np. pogrubienie, pochylenie, podkreślenie tekstu, zmiana kroju i koloru czcionki)		•	•	•	•
stosuje ramki na stronie				•	•
tworzy tabele na stronie, umieszcza w nich tekst, grafikę, poprawnie formatuje komórki tabeli			•	•	•
tworzy własny arkusz stylów na potrzeby wykonanej przez siebie strony				•	•
wykorzystuje skrypty języka JavaScript na stronie					•
<b>Multimedia</b>					
wymienia przykłady plików dźwiękowych i podaje znane programy do ich odtwarzania	•	•	•	•	•
wymienia metody kompresji audio i objaśnia jej zasadę			•	•	•
wymienia formaty plików bitmapowych, zna ich wady i zalety	•	•	•	•	•
zna parametry obrazów rastrowych, rozumie pojęcie przesłoni barw (RGB, CMYK)		•	•	•	•

Umiejętność nabyta przez ucznia	Wymagania				
	K	D	P	R	W
odpowiednio dobiera parametry obrazów i formaty plików do rodzaju publikacji (z uwzględnieniem publikacji dla potrzeb WWW)				•	•
wyjaśnia różnice między grafiką rastrową i wektorową		•	•	•	•
wyjaśnia zasadę tworzenia obrazu płaskiego sceny 3D metodą „śledzenia promienia”				•	•
potrafi wygenerować scenę 3D w poznanym programie			•	•	•
zna zasady tworzenia prezentacji multimedialnej, umie zaprojektować własną prezentację	•	•	•	•	•
przygotowuje materiały w postaci tekstów, rysunków, dźwięków		•	•	•	•
dba o estetykę i czytelność utworzonej prezentacji		•	•	•	•
zna zasady tworzenia animacji			•	•	•
dodaje do prezentacji WWW efekty multimedialne: animację, dźwięk, grafikę			•	•	•
zna zasady publikowania prezentacji w Internecie			•	•	•
<b>Tendencje w rozwoju informatyki i jej zastosowań</b>					
potrafi określić zagrożenia, jakie niesie rozwój technik informatycznych	•	•	•	•	•
omawia historię rozwoju architektury komputerów i systemów operacyjnych	•	•	•	•	•
wskazuje na ogólny kierunek rozwoju technik informatycznych	•	•	•	•	•
orientuje się w najnowszych trendach rozwoju technik informatycznych, ze szczególnym uwzględnieniem usług internetowych		•	•	•	•
potrafi wykorzystywać zasoby sieci w celach edukacyjnych w sposób etyczny, selekcjonuje treść i ocenia jej wiarygodność	•	•	•	•	•
potrafi wymienić najnowsze osiągnięcia w rozwoju technik informatycznych i umie je odnieść do sytuacji sprzed lat			•	•	•

### Metody oceniania uczniów

Proponujemy następujący podział metod oceniania uczniów:

1. Praca i aktywność ucznia na lekcjach.
2. Sprawdziany pisemne bez użycia komputera.
3. Sprawdziany praktyczne przy komputerze.
4. Praca nad projektem grupowym.
5. Współpraca z nauczycielami innych przedmiotów.

6. Materiały (prezentacje, referaty) przygotowywane przez uczniów.
7. Indywidualna praca ucznia poza godzinami lekcyjnymi.
8. Ocena osiągnięć olimpijskich.

### **Praca i aktywność ucznia na lekcjach**

Większość tematów realizowanych na informatyce wymaga zastosowania ścisłego podziału lekcji na poszczególne etapy. Pierwszy etap to wprowadzenie uczniów do tematu, zapoznanie z problemem, który będzie na lekcji realizowany. Nauczyciel przedstawia zadanie, które będzie wykonywane na lekcji, formułuje problem, podaje ewentualne wskazówki konieczne do jego rozwiązania. Uczniowie po zrozumieniu zagadnienia powinni aktywnie i czynnie poszukiwać metod na jego rozwiązanie. Ciekawe i poprawne pomysły uczniów mogą być premiowane ocenami. Kolejnym etapem jest realizacja rozwiązania zadania na tablicy, kartce lub już implementacja na komputerze. Zaangażowanie ucznia w pracę wykonywaną na lekcji również powinno być premiowane ocenami, gdyż ma to motywujące działanie. Uczeń jest świadomy, że na jego ocenę końcową wpływ mają nie tylko oceny cząstkowe uzyskiwane na sprawdzianach, ale również całokształt zaangażowania na każdej lekcji. Lekcja powinna się kończyć etapem podsumowującym pracę, kiedy nauczyciel sprawdza wyniki wszystkich uczniów i nagradza tych, którzy wykonali zadanie poprawnie lub w ciekawy i nowatorski sposób. Ten etap lekcji jest również niezbędny, ze względu na możliwość zaobserwowania, który uczeń nie potrafił sobie z problemem poradzić.

### **Sprawdziany pisemne bez użycia komputera**

Chociaż ucząc informatyki, kładziemy większy nacisk na praktyczne umiejętności, jednak część wiedzy i umiejętności ucznia może być sprawdzana za pomocą sprawdzianów pisemnych lub kartkówek. Sprawdziany pisemne nauczyciel powinien zapowiedzieć odpowiednio wcześniej, zgodnie z regulaminem szkolnym, natomiast kartkówki, czyli krótkie sprawdziany obejmujące materiał z trzech ostatnich lekcji, mogą być niezapowiedziane. Proponujemy, aby zgodnie z nową formułą matury z informatyki stosować sprawdziany podobne w swojej budowie do arkuszy maturalnych. Oprócz uzyskania oceny z takiego sprawdzianu uczeń nabyla umiejętności potrzebne na egzaminie maturalnym.

### **Sprawdziany praktyczne przy komputerze**

Sprawdziany praktyczne to najczęstsza forma sprawdzianów z informatyki. Proponujemy, aby zadania, które uczeń ma wykonać, były zapisane przez nauczyciela na kartkach i rozdane uczniom. Zadania powinny być sformułowane w sposób jasny i jednoznaczny, tak aby uczeń nie musiał zadawać dodatkowych pytań. Dodatkowe wyjaśnianie przez nauczyciela zadań na sprawdzianie jest niezgodne z formułą postępowania na maturze. Odpowiedzi na swoje ewentualne pytania uczeń musi się nauczyć szukać w zadaniu.

### **Praca nad projektem grupowym**

Na informatyce preferujemy podział na dwu-, trzyosobowe grupy. Ocenę pracy grupowej traktujemy jako szczególnie ważną pod względem wychowawczym. Pracując z kolegami w grupie, uczeń ma świadomość, że od jego pracy zależy nie tylko ocena indywidualna, ale również ocena całej grupy. Uczy to odpowiedzialności i umiejętności podziału pracy pomiędzy członków grupy. Od kreatywnego współdziałania i zaangażowania wszystkich uczestników pracy grupowej zależy ocena końcowa projektu.

## **Współpraca z nauczycielami innych przedmiotów**

Proponujemy, aby pozytywną oceną premiować sytuacje, gdy uczeń, korzystając ze swoich umiejętności, wykona pracę (program komputerowy, prezentację multimedialną), która będzie służyć nauczycielowi innego przedmiotu (np. matematyki, fizyki) jako środek dydaktyczny. Przede wszystkim praca taka ma ogromny wpływ motywujący na ucznia, który dostrzega praktyczne wykorzystanie umiejętności zdobytych na lekcji informatyki. Powiększa się też baza środków dydaktycznych szkoły. Najczęstszą sytuacją jest prowadzenie przez uczniów szkolnej witryny internetowej lub strony klasowej. Oprócz faktu, że uczniowie działają dla dobra szkoły, mogą się wykazać wiedzą i umiejętnościami, co nauczyciel powinien nagrodzić pozytywną oceną za pracę dodatkową.

## **Materiały wykonywane przez uczniów oraz indywidualna praca ucznia poza godzinami lekcyjnymi**

Możemy również nagrodzić ucznia oceną za przygotowanie referatu lub prezentacji multimedialnej zgodnej z realizowanym tematem. Taka ocena nie powinna jednak służyć podwyższeniu uczniowi oceny końcowej. Nie należy dopuszczać do sytuacji, kiedy uczniowie zgłaszają chęć przygotowania referatu, licząc na zmianę oceny końcowej. Ocena każdej pracy domowej ucznia wymaga od nauczyciela bardzo dużego wyczucia, ponieważ musimy mieć pewność, że uczeń pracował samodzielnie. Proponujemy, aby prace do wykonania w domu zadawać uczniom zainteresowanym przedmiotem i wykazującym duże zdolności i umiejętności wykraczające zdecydowanie poza program, a zatem potencjalnym kandydatom do oceny celującej. Dla nich nie tyle ważna jest ocena, jaką otrzymają za pracę wykonaną poza godzinami lekcyjnymi, ile satysfakcja z jej wykonania oraz ewentualne uwagi nauczyciela.

## **Ocena osiągnięć olimpijskich**

Osiągnięcia olimpijskie ucznia nie mogą pozostać bez wpływu na jego ocenę końcową. Nie dopuszczamy jednak sytuacji, że nawet wysokie osiągnięcia w olimpiadzie przedmiotowej niejako w automatyczny sposób decydują o bardzo dobrej lub celującej ocenie końcoworocznej. Może się bowiem zdarzyć, że bardzo głębokiej wiedzy i umiejętnościom z jednego działu nie towarzyszą równie ugruntowane wiadomości z innych działów tematycznych obowiązujących w całym roku szkolnym.

## **X. Wymagania egzaminacyjne**

Standardy wymagań, będące podstawą przeprowadzenia egzaminu maturalnego z informatyki obejmują trzy obszary:

- I. Wiadomości i rozumienie.
- II. Korzystanie z informacji.
- III. Tworzenie informacji.

W ramach każdego obszaru cyframi arabskimi i literami oznaczono poszczególne standardy wynikające z podstawy programowej. Przedstawiają one:

- zakres treści nauczania, na których podstawie podczas egzaminu może być sprawdzany stopień opanowania określonej w standardzie umiejętności;

- rodzaje informacji do wykorzystania;
- typy i rodzaje informacji do tworzenia.

Przedstawione poniżej standardy wymagań egzaminacyjnych są dosłownym przeniesieniem fragmentu rozporządzenia Ministra Edukacji Narodowej i Sportu z dnia 10 kwietnia 2003 roku, zmieniającego rozporządzenie w sprawie standardów wymagań, będących podstawą przeprowadzania sprawdzianów i egzaminów.

## **I. Wiadomości i rozumienie**

**Zdający zna i rozumie podstawowe pojęcia, metody, narzędzia i procesy związane z informatyką:**

- 1) opisuje środki, narzędzia i metody informatyki, posługując się poprawną terminologią informatyczną,
- 2) przedstawia rolę, funkcje i zasady pracy sprzętu komputerowego (komputera, urządzeń peryferyjnych, sieci komputerowej),
- 3) charakteryzuje typowe narzędzia informatyczne (oprogramowanie) i ich zastosowania,
- 4) omawia przydatność i wiarygodność różnych źródeł i zbiorów informacji oraz użyteczność sposobów i form ich reprezentowania,
- 5) zna klasyczne algorytmy:
  - a) algorytmy z rozgałęzieniami (np. rozwiązywanie równań liniowych i kwadratowych),
  - b) liniowe przeszukiwanie ciągu w poszukiwaniu wyróżnionego elementu,
  - c) porządkowanie ciągu elementów (metodami: bąbelkową, przez wstawianie, przez wybór, przez scalanie, szybko),
  - d) metoda „dziel i zwyciężaj” (np. przeszukiwanie binarne),
  - e) algorytmy rekurencyjne (np. algorytm Euklidesa, znajdowanie liczb Fibonacciego),
  - f) schemat Hornera,
  - g) algorytmy na liczbach naturalnych (np. pozycyjne reprezentacje liczb, generowanie liczb pierwszych),
  - h) algorytmy numeryczne (np. wyznaczanie miejsca zerowego funkcji, obliczanie wartości pierwiastka kwadratowego),
- 6) opisuje proces rozwoju technologii informacyjnej we współczesnej cywilizacji i rozumie jego znaczenie.

## **II. Korzystanie z informacji**

**Zdający stosuje posiadaną wiedzę do rozwiązywania zadań teoretycznych i praktycznych:**

- 1) posługuje się typowymi programami użytkowymi, takimi jak: edytor tekstu, arkusz kalkulacyjny, program obsługi baz danych, program prezentacyjny, przeglądarka WWW, program do obsługi poczty elektronicznej oraz kompilator wybranego języka programowania,
- 2) rozwiązuje zadania poprzez skorzystanie ze zbioru gotowych rozwiązań,
- 3) wykorzystuje zasoby i usługi sieci komputerowych (komunikację z innymi użytkownikami, przesyłanie danych przez sieć, tworzenie dokumentów dostępnych w sieci),
- 4) stosuje metody wyszukiwania i przetwarzania informacji w relacyjnych bazach danych,

- 5) stosuje klasyczne algorytmy w typowych sytuacjach,
- 6) dobiera właściwy program (użytkowy lub własnoręcznie napisany) do rozwiązywanego zadania,
- 7) zapisuje rozwiązanie zadania w postaci algorytmu ze specyfikacją, w wybranej przez siebie notacji (listy kroków, schematu blokowego, w języku lub pseudojęzyku programowania),
- 8) wykorzystuje zdobytą wiedzę i umiejętności do rozwiązywania zadań z różnych dziedzin (np. z matematyki) i problemów życia codziennego.

### **III. Tworzenie informacji**

#### **Zdający stosuje metody informatyczne do rozwiązywania problemów:**

- 1) formułuje sytuację problemową (w tym podaje specyfikację problemu) i ocenia cechy zaproponowanego rozwiązania,
- 2) formułuje informatyczne rozwiązanie problemu przez dobór odpowiednich struktur danych oraz algorytmu i realizuje je w wybranym języku programowania,
- 3) wykorzystuje metody informatyki (metodę zstępującą, konstrukcje algorytmiczne, klasyczne algorytmy) do rozwiązywania problemu,
- 4) ocenia poprawność i efektywność rozwiązania danego problemu,
- 5) projektuje i tworzy bazy danych będące reprezentacją zbioru informacji i relacji między nimi,
- 6) stosuje narzędzia i techniki informatyczne do modelowania i symulacji procesów oraz zjawisk,
- 7) wykorzystuje różnorodne źródła i zasoby informacji do tworzenia dokumentów tekstowych i multimedialnych,
- 8) formułuje i uzasadnia opinie dotyczące konsekwencji dla osób i społeczeństw, jakie wynikają z zastosowań informatyki i technologii informacyjnej.