ZAKRES CZĘŚCI PODSTAWY PROGRAMOWEJ w zakresie rozszerzonym  
dla liceum ogólnokształcącego  
*Informatyka na czasie*

**Klasa II**

Wymagania do egzaminu z informatyki, przygotowane na podstawie programu nauczania informatyki dla liceum ogólnokształcącego i technikum *Informatyka na czasie. Zakres rozszerzony,* autor Zbigniew Talaga, konsultacja Janusz Mazur.

Nazywam się **Barbara Szlachta**, jestem nauczycielem, absolwentką Uniwersytetu Rzeszowskiego. Ukończyłam studia magisterskie na kierunku matematyka nauczycielska, studia inżynierskie na kierunku informatyka oraz studia podyplomowe z zakresu matematyka w finansach. W roku szkolnym 2021/2022 otrzymałam grant Centrum Mistrzostwa Informatycznego, w ramach którego zdobywałam i poszerzałam swoją wiedzę z informatyki i programowania, biorąc udział w szkoleniach na jednej z pięciu najlepszych uczelni technicznych w Polsce – Akademii Górniczo Hutniczej w Krakowie.

Zapraszam do kontaktu pod adresem: [barbarka\_s@o2.pl](mailto:barbarka_s@o2.pl)

1. **Algorytmy na liczbach całkowitych i tekstach**
2. Od problemu do programu

Uczeń:

* definiuje pojęcie specyfikacja algorytmu, określa dane i wyniki
* planuje kolejne kroki rozwiązania problemu
* omawia różne sposoby przedstawiania algorytmów (opis słowny, lista kroków, schemat blokowy, pseudokod)
* programuje i testuje rozwiązanie problemu
* sprawdza działanie algorytmów dla różnych danych
* tworzy algorytmy działania na liczbach całkowitych
* stosuje w języku C++ podstawowe konstrukcje programistyczne (operacje wejścia i wyjścia, instrukcja warunkowa, operatory matematyczne i logiczne)
* tworzy w języku C++ programy wykonujące działania na liczbach całkowitych

1. Systemy liczbowe i reprezentacja danych w komputerze

* definiuje pojęcie pozycyjnego systemu liczbowego
* wymienia systemy liczbowe stosowane w informatyce
* definiuje pojęcia bit i bajt
* dokonuje konwersji między pozycyjnymi systemami liczbowymi, wykorzystując przy tym zależności między systemami binarnym i ósemkowym oraz binarnym i heksadecymalnym
* omawia sposób reprezentowania liczb całkowitych w komputerze
* wymienia typy danych służące do zapisu liczb całkowitych (short int, int, long int, long long int, unsigned), stosuje je w pisanych programach
* opisuje, jak w komputerze reprezentowane są znaki i napisy (char, string), odwołuje się do znaku w napisie za pomocą indeksu
* wyjaśnia, czym jest tablica kodów ASCII
* omawia działanie operacji logicznych i reprezentację ich wyników w komputerze (wynik może przyjmować wartość prawda – 1 lub fałsz – 0, co zajmuje 1 bajt pamięci)
* opisuje istotę cyfrowej reprezentacji w komputerze obrazów, dźwięków i animacji

1. Algorytmy zamiany reprezentacji liczb między systemami liczbowymi

* tworzy programy do konwersji między liczbami w systemach binarnym i decymalnym
* pisze programy konwertujące liczbę dziesiętną na liczbę w podanym systemie pozycyjnym
* posługuje się środowiskiem programistycznym, strukturami danych oraz językiem programowania w stopniu umożliwiającym implementację omawianych algorytmów
* stosuje binarną reprezentację liczby w algorytmie szybkiego podnoszenia do potęgi

1. Czy to jest palindrom?

* definiuje pojęcie palindromu
* określa, czy dany napis lub liczba są palindromami
* wykonuje operacje na napisach (wczytywanie napisów ze spacjami, sprawdzanie długości napisu, zamiana liter dużych na małe i odwrotnie, porównywanie znaków, znajdowanie oraz usuwanie fragmentów napisów)
* definiuje własne funkcje w języku C++, wyjaśnia celowość ich stosowania, rozróżnia parametry formalne i aktualne
* realizuje w języku C++ algorytmy sprawdzające, czy dany napis jest palindromem, oraz wyszukujące palindromy w zdaniach
* opisuje popularne funkcje oraz metody stosowane dla zmiennych typu string (toupper, tolower, size, substr, erase)

1. Czy ta liczba jest pierwsza?

* wymienia podstawowe własności liczb pierwszych
* sprawdza, czy dana liczba jest pierwsza, stosując algorytm naiwny
* rozkłada liczbę złożoną na czynniki pierwsze
* wyznacza liczby bliźniacze

1. Działania na liczbach w systemach innych niż dziesiętny

* wykonuje działania arytmetyczne na liczbach w różnych systemach pozycyjnych
* wykonuje obliczenia na dowolnie dużych liczbach, wykorzystując napisy
* wyjaśnia różnicę między operacjami na liczbach o podstawie od 1 do 9 i większej od 10
* stosuje odejmowanie w dzieleniu pisemnym liczb binarnych
* stosuje dodawanie liczby przeciwnej zapisanej w kodzie U2 przy odejmowaniu liczby binarnej

1. Algorytm Euklidesa i działania na ułamkach

* opisuje geometryczną interpretację algorytmu Euklidesa
* pisze program realizujący algorytm Euklidesa w wersjach z dzieleniem i odejmowaniem, stosując funkcję typu void
* stosuje strukturę do reprezentacji liczb wymiernych
* wykorzystuje algorytm Euklidesa do działań na ułamkach
* stosuje zmienne lokalne i globalne, a także przekazywanie parametrów przez wartość

1. Szyfr Cezara i inne szyfry podstawieniowe

* definiuje szyfry: podstawieniowy, monoalfabetyczny i permutacyjny, wymienia przykłady takich szyfrów
* pisze program szyfrujący informację szyfrem Cezara z wykorzystaniem liter z polskimi znakami diakrytycznymi
* omawia szyfr Vigenere’a
* stosuje w swoich programach operacje plikowe – wczytywanie danych z pliku dyskowego, zapis wyniku do pliku

1. **Rozwiązywanie problemów z wykorzystaniem struktur danych**
2. Łamiemy szyfr Cezara

* wyjaśnia, na czym polega łamanie szyfru (kryptoanaliza)
* łamie szyfr Cezara, stosując analizę częstości
* stosuje algorytmy zliczające liczbę wystąpień znaków w tekście z zastosowaniem strukturalnego typu danych – tablic
* pisze program znajdujący maksimum w tablicy i wypisujący jego pozycję (algorytm „dziel i zwyciężaj”)

1. Poszukujemy liczby

* znajduje wartość w zbiorach uporządkowanym i nieuporządkowanym, stosując odpowiednio algorytmy wyszukiwania liniowego, liniowego z wartownikiem i binarnego
* pisze programy wykorzystujące przekazywanie parametru do funkcji przez wskaźnik i referencję
* stosuje algorytm „dziel i zwyciężaj” do jednoczesnego znajdowania maksimum i minimum w zbiorze

1. Jak ocenić złożoność obliczeniową algorytmu?

* definiuje złożoność obliczeniową algorytmu
* szacuje złożoność czasową i pamięciową
* wyjaśnia, czym jest złożoność oczekiwana (średnia), optymistyczna i pesymistyczna

1. Metody sortowania prostego

* definiuje pojęcie sortowania, prawidłowo określając klucz i porządek sortowania
* definiuje pojęcia sortowania in situ i stabilnego
* stosuje metody sortowania prostego do sortowania liczb w zbiorze – bąbelkowe i przez wybieranie
* szacuje złożoność obliczeniową stosowanych algorytmów
* definiuje operacje kluczowe (dominujące) w algorytmach sortowania
* pisze programy realizujące poznane algorytmy sortowania

1. Szyfry przestawieniowe, anagramy

* omawia zasadę działania szyfrów przestawieniowych, wymienia przykłady takich szyfrów
* sprawdza, czy słowa (napisy) są anagramami
* pisze funkcje sprawdzające
* wykorzystuje poznane wcześniej algorytmy sortowania i zliczania w rozwiązywaniu problemów

1. Sito Eratostenesa

* opisuje algorytmy sprawdzające, czy liczba jest pierwsza
* omawia i stosuje algorytm sita Eratostenesa do wyszukiwania liczb pierwszych w określonym przedziale liczbowym
* określa złożoność obliczeniową algorytmu

1. Szukamy różnych podciągów

* definiuje pojęcia podciągu oraz podciągu spójnego
* znajduje w zbiorze podciągi o różnych własnościach
* oblicza długość najdłuższego niemalejącego spójnego podciągu oraz liczbę jego elementów
* wymienia i stosuje różne algorytmy znajdowania maksymalnej sumy elementów spójnych podciągów, oceniając ich złożoność obliczeniową
* znajduje w zbiorze spójny podciąg o maksymalnej sumie i wypisuje jego elementy

1. W poszukiwaniu lidera i idola

* definiuje pojęcia idola w grupie i lidera w zbiorze
* znajduje idola w grupie lub stwierdza jego brak
* określa, czy w zbiorze jest lider
* omawia i implementuje w języku C++ algorytmy szukania idola oraz lidera
* ocenia złożoność obliczeniową stosowanych algorytmów i ich efektywność
* stosuje tablice dwuwymiarowe w pisanych programach
* stosuje funkcję sort z biblioteki STL do wyszukiwania lidera

1. **Metody algorytmiczne**
2. Iteracja a rekurencja

* opisuje zasadę działania rekurencji
* implementuje w języku C++ algorytmy rekurencyjne, określa warunki brzegowe
* porównuje iteracyjne i rekurencyjne wersje algorytmów
* opisuje zasadę złotego podziału
* oblicza n-ty wyraz ciągu Fibonacciego metodami iteracyjną i rekurencyjną
* wyjaśnia, na czym polega rozszerzony algorytm Euklidesa, oraz implementuje go w języku C++

1. Metoda zachłanna

* wyjaśnia, na czym polega metoda zachłanna, i wymienia przykłady jej stosowania
* implementuje następujące algorytmy zachłanne: problem kasjera (wydawania reszty minimalną liczbą nominałów), problem telewidza/kinomana (optymalny harmonogram wykorzystania sali), pakowanie plecaka, wyszukiwanie optymalnej drogi
* ocenia przydatność zastosowanych algorytmów
* stosuje własne kryterium porównania w funkcji sort z biblioteki STL

1. Programowanie dynamiczne

* wyjaśnia, na czym polega metoda dynamiczna
* implementuje optymalne algorytmy dotyczące problemu kasjera, telewidza, znajdowania drogi oraz pakowania plecaka
* stosuje metodę dynamiczną do znajdowania najdłuższego wspólnego podciągu
* porównuje metody zachłanną i dynamiczną

1. Dziel i zwyciężaj, czyli sortujemy sprawniej

* omawia metodę „dziel i zwyciężaj” oraz rekurencję
* wyjaśnia, na czym polega algorytm sortowania szybkiego oraz przez scalanie i implementuje je
* ocenia i porównuje złożoność czasową i obliczeniową algorytmów