

## **Wymagania edukacyjne z informatyki**

### **KLASA 3 poziom rozszerzony**

#### **Algorytmy na liczbach całkowitych**

Ocenę *bardzo dobrą* otrzymuje uczeń, który:

- *pisze programy stosując pętle zagnieżdżone,*
- *stosuje złożone warunki logiczne w instrukcjach sterujących,*
- *umie analizować poprawność tworzonych programów,*
- *rozumie pojęcie złożoności obliczeniowej programu.*

Ocenę *dobłą* otrzymuje uczeń, który:

- *określa specyfikację algorytmu (dane, wynik),*
- *pisze programy o różnym stopniu trudności, szacuje ich efektywność,*
- *przedstawia omawiane algorytmy w postaci opisu słownego, listy kroków, schematu blokowego, pseudokodu,*
- *dobiera typy danych do realizacji problemu,*
- *umie zastosować iteracje do przetwarzania danych wejściowych,*
- *umie zastosować i poprawnie wypisać typ rzeczywisty*

Ocenę *dostateczną* otrzymuje uczeń, który:

- *definiuje pojęcie zdania logicznego, charakteryzuje podstawowe operacje logiczne*
- *(koniunkcja, alternatywa, negacja) oraz operatory logiczne,*
- *pisze programy wykonujące działania na liczbach całkowitych,*

Ocenę *dopuszczającą* otrzymuje uczeń, który:

- *definiuje podstawowe pojęcia z algorytmiki i programowania: algorytm, program, warunek,*
- *iteracja, rekurencja,*
- *wymienia sposoby reprezentacji algorytmów,*
- *korzysta ze środowiska programistycznego: pisze w nim kod, kompiluje i uruchamia program,*
- *pisze programy o niewielkim stopniu trudności,*

#### **Metody algorytmiczne**

Ocenę *bardzo dobrą* otrzymuje uczeń, który:

- *projektuje algorytm badający właściwości liczb (podzielność, pierwszość, cyfry) w podanym przedziale liczb naturalnych przy pomocy schematu blokowego lub pseudokodu,*
- *projektuje algorytm generujący n kolejnych liczb naturalnych o zadanych właściwościach (podzielność, pierwszość, cyfry) w podanym przedziale liczb naturalnych przy pomocy schematu blokowego lub pseudokodu.*

Ocenę *dobłą* otrzymuje uczeń, który:

- *projektuje algorytm rozkładu liczby na czynniki pierwsze przy pomocy schematu blokowego lub pseudokodu,*
- *projektuje algorytm zliczający sumę cyfr z jakich składa się liczba przy pomocy schematu blokowego lub pseudokodu,*
- *projektuje algorytm zliczający sumę liczb w podanym ciągu przy pomocy schematu blokowego lub pseudokodu,*

Ocenę *dostateczną* otrzymuje uczeń, który:

- *definiuje pojęcie zdania logicznego, charakteryzuje podstawowe operacje logiczne (koniunkcja, alternatywa, negacja) oraz operatory logiczne,*
- *pisze programy wykonujące działania na liczbach całkowitych,*
- *projektuje algorytm naiwny sprawdzający, czy liczba jest pierwsza przy pomocy schematu blokowego lub pseudokodu,*

Ocenę *dopuszczającą* otrzymuje uczeń, który:

- *definiuje podstawowe pojęcia z algorytmiki i programowania: algorytm, program, warunek,*

- iteracja, rekurencja,
- wymienia sposoby reprezentacji algorytmów,
- korzysta ze środowiska programistycznego: pisze w nim kod, kompiluje i uruchamia program, odczytuje i zapisuje pliki,
- projektuje programy o niewielkim stopniu trudności przy pomocy schematów blokowych lub pseudokodu,

### **Algorytmy na tekstach**

Ocenę bardzo dobrą otrzymuje uczeń, który:

- stosuje różne sposoby przekazywania parametrów do funkcji, uzasadnia ich użycie,
- pisze funkcje typu logicznego, np. sprawdzającą, czy napis jest palindromem,
- sprawdza, czy napisy są anagramami, stosując sortowanie lub zliczanie znaków,
- pisze program odczytujący informację ukrytą za pomocą szyfru Cezara z wykorzystaniem analizy częstości znaków w tekście,
- stosuje funkcję haszującą oraz algorytm Karpa–Rabina w programach wyszukujących wzorzec w tekście,

Ocenę dobrą otrzymuje uczeń, który:

- implementuje algorytmy sprawdzające, czy napis jest palindromem,
- pisze programy sprawdzające, czy dwa napisy są anagramami,
- szyfruje dane wczytane z pliku tekstowego,
- implementuje algorytm zliczania znaków w tekście oraz wyszukujący maksimum z wykorzystaniem tablic,
- omawia algorytm Karpa–Rabina do wyszukiwania wzorca w tekście z zastosowaniem funkcji haszującej,

Ocenę dostateczną otrzymuje uczeń, który:

- korzysta z biblioteki string do operacji na łańcuchach znaków,
- wykonuje operacje na napisach, wykorzystując słowa kluczowe: size, find, substr, erase, toupper, tolower,
- wczytuje napisy ze spacjami, wykorzystując słowo kluczowe getline,
- tworzy algorytmy sprawdzające, czy napis jest palindromem,
- przedstawia w postaci algorytmu problem wyszukiwania anagramów,
- pisze program szyfrujący napis szyfrem Cezara,
- implementuje algorytm naiwny wyszukiwania wzorca w tekście,
- wyjaśnia metodę haszowania,

Ocenę dopuszczającą otrzymuje uczeń, który:

- wyjaśnia, czym jest tablica kodów ASCII,
- wyjaśnia, czym są palindrom i anagram, podaje przykłady,
- omawia szyfr Cezara jako przykład szyfru podstawieniowego i szyfr kolumnowy jako przykład szyfru przestawieniowego,
- omawia algorytm zliczania znaków w tekście,
- wyszukuje wzorzec w tekście algorytmem naiwnym,
- rozumie działanie funkcji haszującej,