**LISTA 1-1A: 4.5**

**LISTA 1-2A: 3.0** (ze względu na braki i istotne błędy w zadaniu z OOP)

**KOMENTARZE – lista 1-1A**

Ogólne:

1. Z punktu widzenia czytelności kodu, zamykanie warunków w instrukcjach *if…elif…else* w nawiasach (zwłaszcza jeżeli nie ma to na celu rozdzielenia bloków połączonych operatorami logicznymi) jest mylące, ponieważ sprawia, że instrukcja wygląda jak wywołanie funkcji – w Pythonie dominuje styl bez nawiasów, np. w zadaniu 1:

|  |
| --- |
| Aktualna wersja |
| 6 if (len(args)==4): |
| Wersja alternatywna |
| 6 if len(args)==4: |

1. Style guide dla Pythona określa, że nazwy funkcji powinny być pisane w całości małymi literami ze słowami rozdzielanymi podkreślnikiem (np. w zadaniu 3 powinno być raczej *multiplication\_table()* niż *MultiplicationTable*), CamelCase i podobne style są raczej zarezerwowane dla nazw klas.
2. W Pythonie 3 zalecanym podejściem do formatowania stringów jest używanie funkcji *str.format()* zamiast notacji z operatorem % albo łączenia stringów przez znak „+” (m.in. dlatego, że np. rozwiązuje problem konwersji na stringi).
3. W przypadku z góry określonej liczby iteracji pętli do wykonania (tak jest w większości zadań z tej listy) sensowniej jest wykorzystać pętlę *for* albo odpowiednik pętli *foreach* niż pętlę *while*.
4. W przypadku wystąpienia wartości nie pozwalających na obliczenie odpowiedniej liczby elementów (np. mniejszej lub równej 0) poza wyświetleniem komunikatu sensowniej byłoby od razu wychodzić z programu za pomocą funkcji *sys.exit()* albo analogicznej niż polegać na zabezpieczeniu przez pętlę *while*, która nie wykona się ani razu.

Zadanie 1

1. PROBLEM: Sprawdzanie warunku na istnienie trójkąta nie działa – wartość zwracana z funkcji *sorted()* nie jest przypisywana do żadnej zmiennej, przez co w bloku *if* sprawdzana jest zawartość zmiennej *table* bez sortowania – nawet gdyby sortowanie było wykonywane „in place”, to wywołanie miało by postać *table.sorted()*, a nie samo *sorted(table)*. Jeżeli warunek nie jest spełniony, to z funkcji *math.sqrt()* rzucany jest wyjątek typu *ValueError: math domain error*, ale dalszy blok *except* przechwytuje ten typ wyjątków i wyświetla go z nieprawidłowym komunikatem o błędnym typie argumentów. Ponadto sortowanie nie działa zgodnie z założeniem, ponieważ domyślnie wartości są sortowane w kolejności rosnącej, a warunek jest napisany dla kolejności malejącej.

|  |
| --- |
| Aktualna wersja |
| 12 table = [a, b, c]  13 sorted(table)  14  15 if table[0] < (table[1] + table[2]): |
| Wersja alternatywna |
| 12 table = [a, b, c]  13 table = sorted(table, reverse=True)  14  15 if table[0] < (table[1] + table[2]): |
|  |

Zadanie 2

1. Linia 14 – w Pythonie nie kończy się linii średnikiem.

Zadanie 4

1. PROBLEM: Wynik jest wypisywany w nieprawidłowym formacie - z treści zadania: „(wyraz): (liczba\_samogłosek)”.
2. Zawartość zmiennej *sys.argv* to stringi, nie ma potrzeby ponownej konwersji przed użyciem metody *lower()*.

Zadanie 5

1. Zamiast stosować dwa zagnieżdżone bloki *if…else* można zastosować jeden blok *if…elif…else –* pierwsze dwa warunki są równorzędne, a poprawiłoby to czytelność kodu.
2. W przypadku obliczania wzajemnego położenia prostych na podstawie współczynników wystarczyłoby podanie współczynników kierunkowych (*a1*,*a2*), a nie pełnego równania prostej – zmienne *b1*, *b2* są nieużywane.

Zadania 6-8:

1. PROBLEM: Przy odczycie z pliku brakuje sprawdzania, czy odczytana wartość temperatury jest liczbą całkowitą – jeżeli podana została liczba zmiennoprzecinkowa lub string, to zostanie rzucony wyjątek typu *ValueError*, który jest przechwytywany przez dalszy blok *except* i wyświetlany z nieprawidłowym komunikatem o błędnym typie argumentów.
2. PROBLEM: W wersji z wydzielonym modułem brakuje zamykania pliku otwartego do zapisu.
3. PROBLEM: W zadaniu 8 dwa zagnieżdżone bloki *if…else* można zastosować jeden blok *if…elif…else –* warunki na tryb pracy są równorzędne, a poprawiłoby to czytelność kodu. W aktualnej wersji blok *else* znajduje się w nieprawidłowym miejscu, przez co komunikat o nieprawidłowym trybie pracy i nie trollowaniu nigdy nie zostanie wyświetlony.
4. Zawartość zmiennej *sys.argv* to stringi, nie ma potrzeby ponownej konwersji.
5. Należy unikać nazywania zmiennych czy funkcji nazwami typów i metod wbudowanych (tutaj: *list*, *str*).
6. Brakuje formatowania wyświetlanego wyniku – wypisywanie surowej tablicy jest mało czytelne.

**KOMENTARZE – lista 1-2A**

Zadanie 1

Nie mam co oceniać, „sorry”.

Zadanie 2

1. PROBLEM: Brakuje sprawdzania, czy podana ścieżka do folderu istnieje (np. przez przechwytywanie wyjątku typu *FileNotFoundError*, który może zostać rzucony z linii 20). Analogicznie brakuje komunikatu w sytuacji, gdy w podanym folderze nie ma pliku o podanym rozszerzeniu.
2. Zawartość zmiennej *sys.argv* to stringi, nie ma potrzeby ponownej konwersji.

Zadanie 3

1. PROBLEM: W przypadku przechwycenia wyjątku z metodzie *\_\_init\_\_()* lepiej byłoby „przerzucić” ten wyjątek dalej niż polegać na komunikacie wyświetlanym w konsoli (w przypadku klas nie ma podstaw zakładać, że standardowe wyjście będzie dostępne np. jako część większego programu, ponadto chodzi raczej o uniemożliwienie stworzenia obiektu z nieprawidłowymi współczynnikami).
2. PROBLEM: Jeżeli na końcu listy współczynników są podane zera, to nie powinny one być usuwane – w przeciwnym razie stopień wielomianu jest sztucznie zaniżany, bo te usuwane zera sygnalizują wyrazy, które mają zerowe współczynniki.
3. PROBLEM: Brakuje implementacji mnożenia wielomianów.
4. PROBLEM: Przy odejmowaniu wielomianów nie jest prawidłowo uwzględniana sytuacja, gdy wielomian odejmowany jest dłuższy i współczynniki najwyższych stopni powinny mieć zmienione znaki.
5. PROBLEM: Przy wyświetlaniu reprezentacji wielomianu brakuje rozróżnienia na dodatnie i ujemne wartości współczynników (przy ujemnych współczynnikach wyświetlany jest zarówno znak „+”, jak „–„).