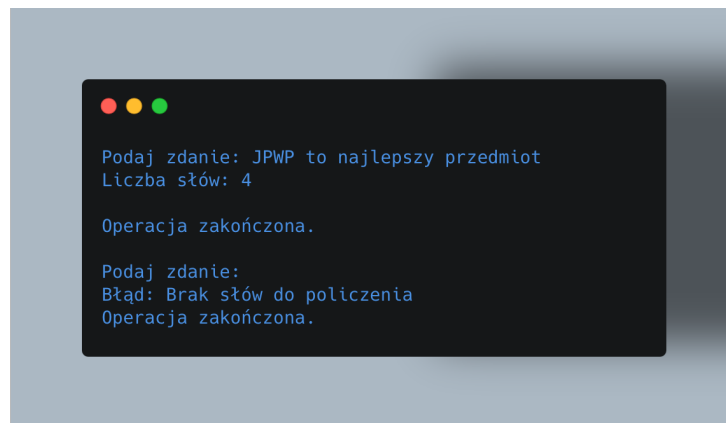


Zaawansowana obsługa wyjątków i debugowanie - treści zadań

Zadanie 1

Twoim zadaniem jest zdefiniować własny wyjątek `EmptyInputError`, dodać walidację w funkcji `count_words`, złapać utworzony wcześniej wyjątek i wyłapać błąd w funkcji `main()`. Aby sprawdzić prawidłowe rozwiązanie zadania uruchom program, następnie wpisz zdanie „*JPWP to najlepszy przedmiot*” i sprawdź, co wypisze program, w kolejnym kroku zostaw lukę pustą (naciśnij enter) i sprawdź co wypisze program.

Oczekiwany wynik:



```
Podaj zdanie: JPWP to najlepszy przedmiot
Liczba słów: 4

Operacja zakończona.

Podaj zdanie:
Błąd: Brak słów do policzenia
Operacja zakończona.
```

Zadanie 2

W tym zadaniu mamy prosty program kalkulatora, twoim zadaniem jest zdefiniowanie dwóch wyjątków:

- `UnsupportedOperationError` — ma być rzucany, gdy podany operator nie jest jednym z dozwolonych: `+`, `-`, `*`, `/`
- `InvalidNumberError` — ma być rzucany, gdy próba konwersji tekstu na liczbę zmiennoprzecinkową (`float`) się nie powiedzie

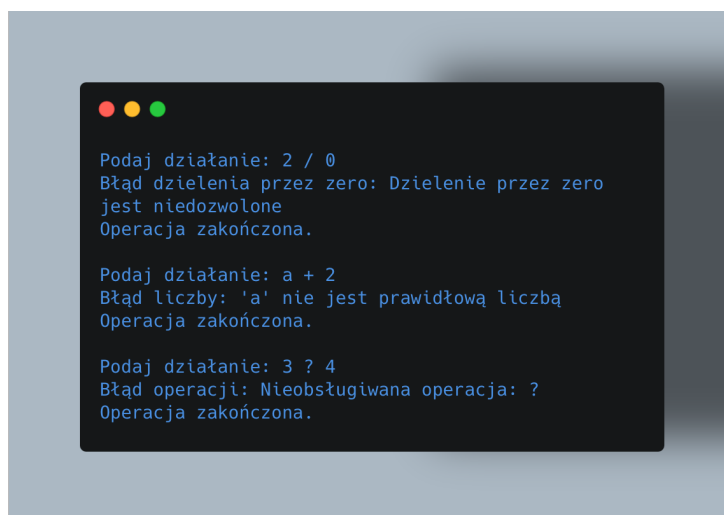
Następnie uzupełnij funkcję `safe_convert_to_float()` tak aby próbowała zmienić zmienną `s` na `float`, jeśli konwersja się nie uda, funkcja ma rzucić nasz własny wyjątek.

W kolejnym kroku w funkcji `calculate()` dodaj:

- Korzystanie z funkcji `safe_convert_to_float()`
- Obsługę dzielenia przez zero
- Obsługę naszego wyjątku `UnsupportedOperationError`

Na sam koniec w funkcji `main()` uzupełnij jeszcze łapanie naszych wyjątków i uruchom program.

Oczekiwany wynik:



```
Podaj działanie: 2 / 0
Błąd dzielenia przez zero: Dzielenie przez zero
jest niedozwolone
Operacja zakończona.

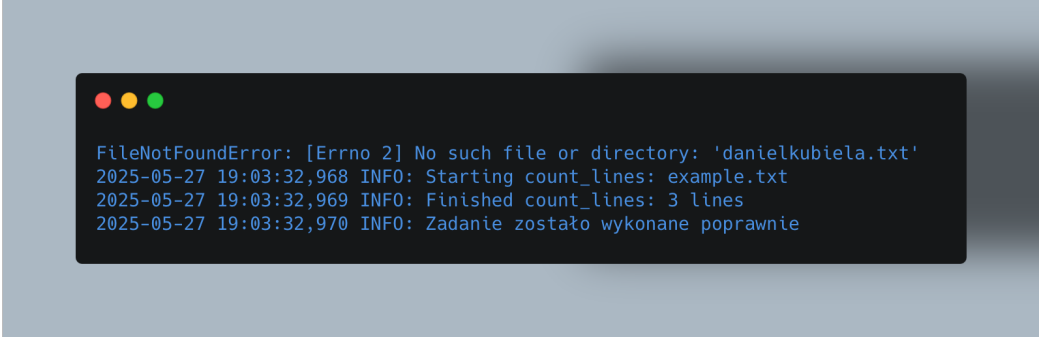
Podaj działanie: a + 2
Błąd liczby: 'a' nie jest prawidłową liczbą
Operacja zakończona.

Podaj działanie: 3 ? 4
Błąd operacji: Nieobsługiwana operacja: ?
Operacja zakończona.
```

Zadanie 3

Program polega na zliczaniu liczby linii w pliku tekstowym. Do wykonania tego zadania potrzebny ci będzie plik `example.txt` z naszego repozytorium. Twoim zadaniem jest skonfigurować logger w funkcji `setup_logger()`, uzupełnić funkcję `count_lines()` tak, aby logowała komunikaty do naszego loggera, jeśli plik nie istnieje ma logować error o tym, że plik nie istnieje, jeśli natomiast został podany poprawny plik ma zalogować info z ilością policzonych wierszy oraz podać komunikat, że zadanie zostało wykonane poprawnie. Jako wynik prześlij komunikaty z loggera.

Oczekiwany wynik:

A screenshot of a terminal window with a dark background and light blue text. The window has three colored window control buttons (red, yellow, green) in the top-left corner. The log output shows an initial error message, followed by two informational messages about starting and finishing the line count for 'example.txt', and a final informational message stating the task was completed successfully.

```
FileNotFoundError: [Errno 2] No such file or directory: 'danielkubiela.txt'
2025-05-27 19:03:32,968 INFO: Starting count_lines: example.txt
2025-05-27 19:03:32,969 INFO: Finished count_lines: 3 lines
2025-05-27 19:03:32,970 INFO: Zadanie zostało wykonane poprawnie
```

Zadanie 4

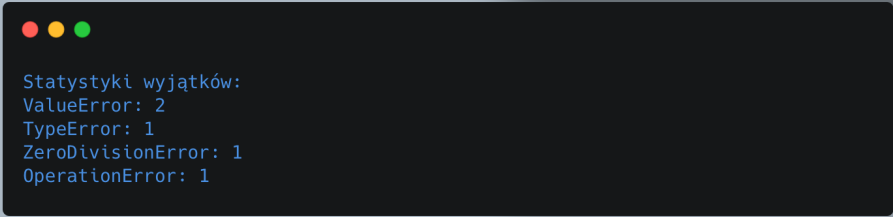
W tym zadaniu należy uzupełnić funkcję `process_item(item)` oraz `process_items(items)`. Funkcja `process_item(item)` musi:

- rzucić wyjątek `ValueError`, jeśli `item` to "value",
- rzucić wyjątek `TypeError`, jeśli `item` to "type",
- rzucić wyjątek `ZeroDivisionError`, jeśli `item` to "zero",
- rzucić własny wyjątek `OperationError`, jeśli `item` to "custom",
- lub zwrócić komunikat, że element został przetworzony.

Następnie uzupełnij funkcję `process_items(items)` tak, aby:

- wylapywała wszystkie wymienione wyjątki,
- zliczała, ile razy wystąpił każdy z nich,
- wypisywała wynik przetwarzania elementu lub pomijała wypisywanie, jeśli wystąpił wyjątek,
- na końcu zwracała słownik z liczbą wystąpień każdego wyjątku.

Oczekiwany wynik:



```
Statystyki wyjątków:  
ValueError: 2  
TypeError: 1  
ZeroDivisionError: 1  
OperationError: 1
```