Wyjątki

Podstawy programowania w języku Python

Błędy i sytuacje wyjątkowe

- uniknięcie błędów podczas tworzenia programów jest prawie niemożliwe
- część błędów to błędy programistyczne, zazwyczaj można je wykryć stosując odpowiednie techniki testowania oprogramowania
- istnieją błędy, których nie da się uniknąć na etapie pisania kodu, np. wynikają z niedoskonałości przyszłych danych
- powyższe sytuacje nazywają się wyjątkami (exceptions)

```
Traceback (most recent call last):
  File "/PycharmProjects/python/tmp.py", line 1, in <module>
    print(1/0)
ZeroDivisionError: division by zero
```

Błąd składni kontra wyjątek

SyntaxError

Błąd składni występuje, gdy próbujemy uruchomić składniowo nieprawidłowy kod Pythona. Taki kod w ogóle nie jest wykonywany, nawet te linie, które są poprawne składniowo.

Exception

Wyjątek występuje, gdy podczas wykonywania składniowo poprawnego kodu Pythona wystąpi błąd. W terminologii Pythona zgłaszany jest wyjątek.

Występowanie wyjątków

- pomimo najlepiej napisanego kodu, zdarzają się sytuacje, że błąd jest nie do uniknięcia
- najczęściej związane z danymi lub parametrami wprowadzanymi do programu z zewnątrz
- najczęściej te sytuacje są nie do uniknięcia, bo nie wiemy co z napisanym przez nas kodem będzie się działo
- sytuacje, w których program musi wykonać operację, która nie jest możliwa do wykonania, prowadzi najczęściej do zakończenia działania skryptu i wypisania komunikatu o błędzie
- w ten sposób język Python informuje nas, że “coś poszło nie tak”

Często spotykane typy wyjątków

- TypeError - nie można wykonać operacji w ramach danego typu, np. dodać wartości do napisu
- IndexError - przekroczenie zasięgu listy lub krotki
- ValueError - nie obsługiwana wartość lub typ danych
- ZeroDivisionError - błąd dzielenia przez zero

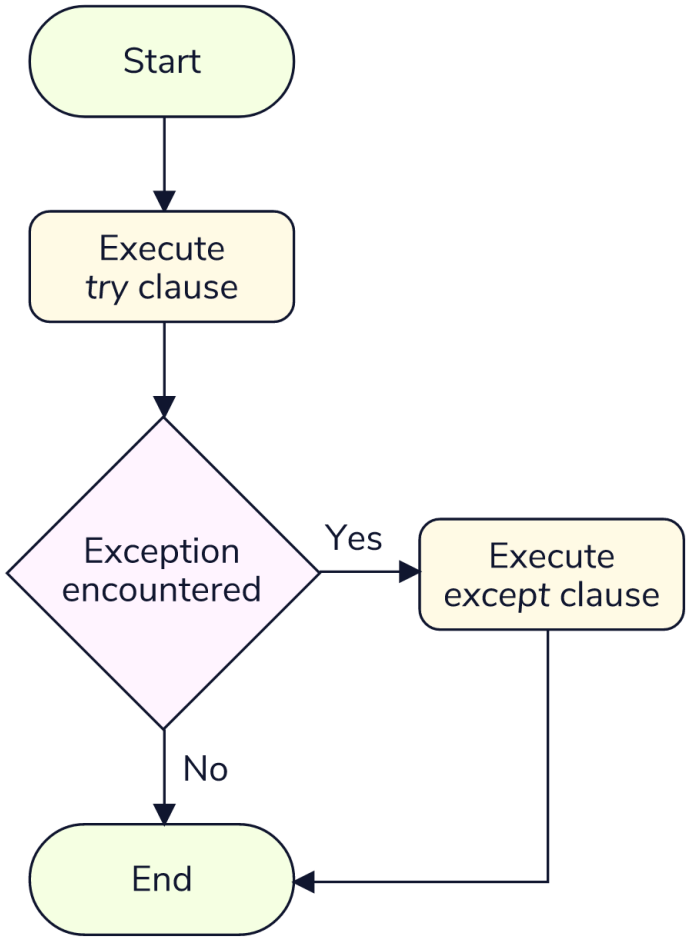
Obsługa wyjątków

- dobrze napisany kod powinien posiadać obsługę wyjątków
- należy przewidzieć, że w danym miejscu kodu może dojść do wyjątku
- fragment kodu, który może wygenerować wyjątek, można próbować obsłużyć przy pomocy konstrukcji **try...except**

```
1  try:
2      x = 0
3      print(1 / x)
4  except:
5      print("Nie wolno dzielić przez zero!")
```

Konstrukcja try…except

- każda podejrzana instrukcja powinna znajdować się w bloku **try:**
- jeżeli gdzieś w bloku **try:** wystąpi wyjątek, to wykonywanie tego fragmentu kodu zostanie przerwane a sterowanie zostanie przekierowane do bloku **except:**
- po wykonaniu kodu z bloku **except:** program będzie kontynuował działanie (od końca bloku except)



Obsługa wielu wyjątków

```python
try:
    number = int(input("Podaj liczbę całkowitą: "))
    print("Odwrota liczba to", 1 / number)
except ValueError:
    print("To nie jest liczba całkowita")
except ZeroDivisionError:
    print("Błąd dzielenia przez zero")
except:
    print("Coś poszło nie tak...")
```

- można użyć wielu bloków **except:** wewnątrz jednej instrukcji **try:** i określić konkretne nazwy wyjątków
- jeśli jedna z gałęzi **except:** zostanie wykonana, pozostałe gałęzie zostaną pominięte
- określony wyjątek wolno wskazać tylko raz
- ogólny wyjątek (ten bez podanej nazwy), powinien być umieszczony na dole gałęzi

Pytanie

Jak powinniśmy obsłużyć błąd SyntaxError?

a) należy zastosować konstrukcję try…except
b) to wyjątek, którego nie powinno się obsługiwać i należy go zignorować
c) błąd składni, powinien zostać wyłapany i poprawiony w fazie testowania
d) w tym przypadku należy zastosować instrukcję warunkową if

Odpowiedź: c)

Błędy składni powinny zostać zdiagnozowane i poprawione w fazie programowania i testowania oprogramowania. Tego typu błędów nie powinno się obsługiwać za pomocą konstrukcji try…except, bo wiadomo jak ich uniknąć.

Pytanie

Kiedy zostanie wykonany kod z bloku **except:** w konstrukcji **try…except**?

a) ten fragment wykonuje się zawsze, niezależnie od wystąpienia wyjątku
b) tylko jeżeli wystąpi wyjątek w bloku **try:**
c) to zależy od typu wyjątku jaki wystąpi, dla wyjątku typu **TypeError** nigdy się nie wykona

Odpowiedź: b)
Należy jednak pamiętać, że można użyć wielu bloków **except:** dla jednej instrukcji **try:** i określić konkretne nazwy wyjątków. Jeśli jedna z gałęzi **except:** zostanie wykonana, pozostałe gałęzie zostaną pominięte.

Pytanie

Co wyświetli się na ekranie po wykonaniu poniższego skryptu, jeżeli użytkownik wpisze 0?

```
a)  Błędna wartość…
b)  Nieprawidłowa wartość…
c)  Błąd!
d)  0
e)  ValueError: …
```

```
1   try:
2       value = int(input("Wprowadź liczbę: "))
3       print(value/value)
4   except ValueError:
5       print("Błędna wartość...")
6   except ZeroDivisionError:
7       print("Nieprawidłowa wartość...")
8   except:
9       print("Błąd!")
```

Odpowiedź: b)

Wystąpi wyjątek dzielenia przez zero (ZeroDivisionError). Wyjątek zostanie obsłużony a użytkownik zobaczy napis ”Nieprawidłowa wartość…”.