

PODSTAWY PROGRAMOWANIA W PYTHON

Dzień 8



AGENDA

DAY 4

- debugowanie
- wyjątki
- refaktoryzacja

1. Debugowanie

DEBUGGING

Odnajdywanie i usuwanie błędów z programu

Bug – ang. robak

THERE'S A BUG IN MY COMPUTER?



BETTER SPRAY SOME RAID ON IT

1947 – Admiral Grace Hopper

9/9

0800 Antan started
1000 " stopped - antan ✓

1300 (032) HP-MC 1.58267000 9.037847025
2.130476415 (2) 4.6159250
(033) PRO 2 2.130476415
convd 2.130676415

Relays 6-2 in 033 failed special speed test
in relay 11.00 test.

Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Multi-Adder Test.

1545

Relay #70 Panel F
(moth) in relay.

First actual case of bug being found.

1630 Antan started.
1700 closed down.



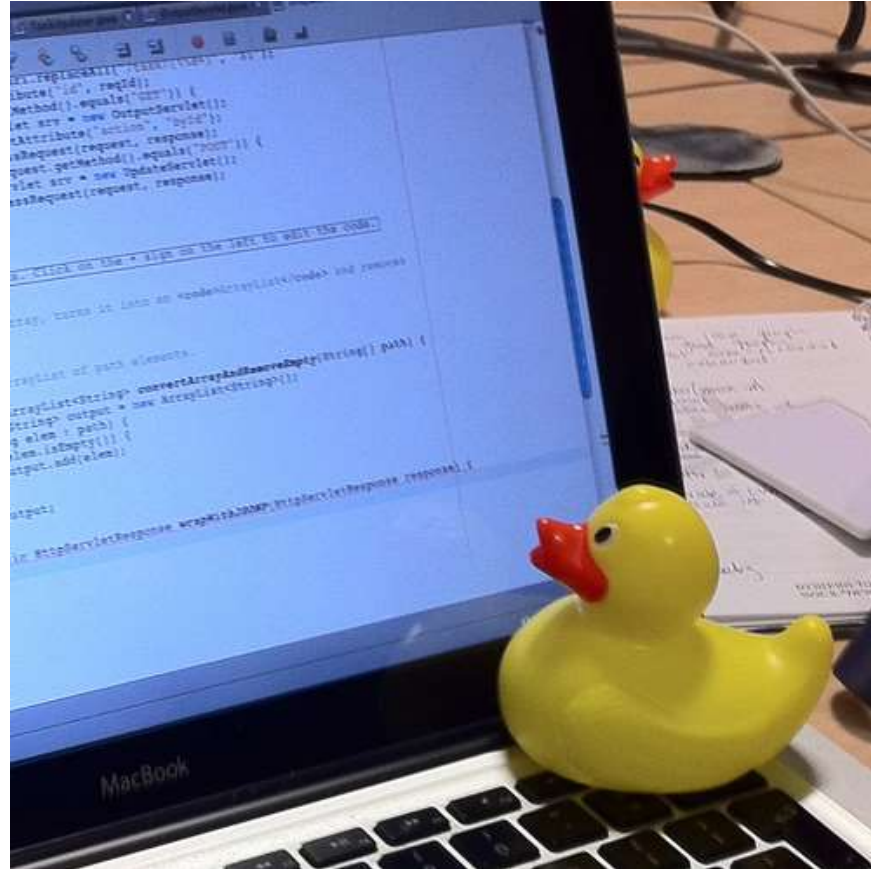


Margaret Hamilton

DEBUGGING

- debugowanie za pomocą print()
- debugowanie za pomocą debuggera
- debugger – program / moduł ułatwiający debuggowanie
 - zatrzymanie programu w dowolnym miejscu
 - wykonywanie instrukcji krok po kroku
 - podgląd aktualnego stanu zmiennych, pamięci

RUBBER DUCK DEBUGGING



Wyjątki

Exceptions

EXCEPTIONS

WYJĄTKI

Syntax Error = błąd w składni polecenia

Exception, Error = Wyjątki – to inaczej błędy powstałe w trakcie wykonywania programu

np. błąd dzielenia przez zero, brak zdefiniowanej zmiennej, odwołanie się do nie istniejącego indeksu itp.

Wyjątek można samemu wywołać celowo!

OBSŁUGA WYJĄTKÓW

używamy bloków try, except, finally

try:

kod

except *TypBłędu*:

kod wywołany gdy zostanie przechwycony błąd

o danym typie

finally: *nieobowiązkowy*

kod wywoływany zawsze, gdy błędu nie będzie

i gdy będzie

WYJĄTKI

Staramy się wyłapywać konkretne typy wyjątków, np. **ValueError**, **NameError** zamiast ogólnego wyjątku **Exception**.

bloki except deklarujemy od szczegółu do ogółu

W przypadku kilku błędów, tylko jeden blok except zostanie wywołany, ten najwyżej.

Refektoryzacja

REFAKTORYZACJA

Proces poprawiania struktury kodu, bez zmiany jego funkcjonalności.





Thanks!!