PODSTAWY PROGRAMOWANIA W PYTHON

Dzień 8

AGENDA
DAY 8

- debugowanie
- wyjątki
- refaktoryzacja
- Python w praktyce

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
imgflip.com

9/9

0800 Arctan started
1000 " stopped - arctan ✓ { 1.2700 9.037 847 025
9.037 846 795
13"uc (032) MP - MC ~~2.130476915~~) 4.6159250
(033) PRO 2 2.130476915
correct 2.130676915
Relays 6-2 in 033 failed special speed test
tm relay 11.00 test.
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Multy Adder Test.
1545  Relay #70 Panel F
(moth) in relay.
First actual case of bug being found.
1700 closed down.

1947 – Admirał Grace Hopper

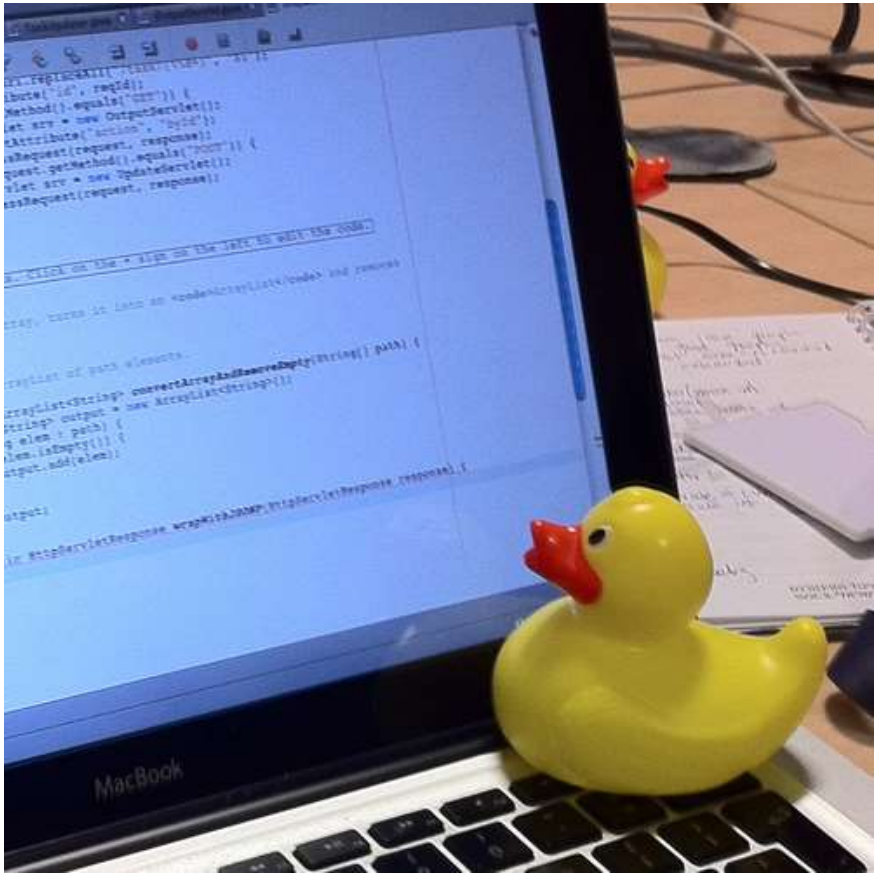
Margaret Hamilton

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:
    # kod wywoływany zawsze, gdy błędu nie będzie
    # i gdy będzie
```

nieobowiązkowy

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.

Wyzwalanie wyjątków

Dzięki użyciu polecenia raise możemy sami wywołać wyjątek

```
raise ValueError("To jest komunikat")
```

Refektoryzacja

REFAKTORYZACJA

Proces poprawiania struktury kodu, bez zmiany jego funkcjonalności.

Python praktycznie

You’ve got mail

wysyłanie email

moduły

smtplib – Simple Mail Transfer Protocole

imaplib – obsługa poczty IMAP

email.mime.MimeText – format przesyłania informacji MIME

https://docs.python.org/3.1/library/email-examples.html
https://docs.python.org/3/library/smtplib.html#module-smtplib

wysyłanie email
konfiguracja gmail

Serwer poczty przychodzącej (IMAP):	imap.gmail.com Requires SSL:Yes (Wymaga połączenia SSL: Tak) Port: 993
Serwer poczty wychodzącej (SMTP):	smtp.gmail.com Requires SSL:Yes (Wymaga połączenia SSL: Tak) Requires TLS: Yes (if available) (Wymaga połączenia TLS: Tak (jeśli jest dostępne)) Requires authentication: Yes (Wymaga uwierzytelnienia: Tak) Port na potrzeby połączeń SSL: 465 Port na potrzeby połączeń TLS/STARTTLS: 587
Imię i nazwisko lub Nazwa wyświetlana	Imię i nazwisko
Nazwa konta, Nazwa użytkownika lub Adres e-mail	Twój pełny adres e-mail
Hasło	Twoje hasło do Gmaila

PSEUDOKOD

importuje biblioteki

mam temat wiadomości i treść wiadomości

tworze mailera

witam się z serwerem smtp – tworzę połaczenie

włączam szyfrowanie (bo chce przesłać do serwera login i hasło)

loguję się (podając login i hasło)

wysyłam maila

kończę połączenie z serwerem

Thanks!!