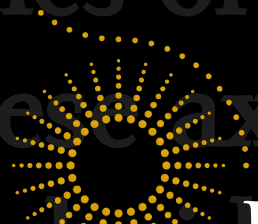
Funkce & Cykly
(6. přednáška)



FF UK
Department of Logic

Jonathan L. Verner

while cykly

FF UK
Department of Logic

```
while <condition>:
  BLOKA
BLOKB
```

- pokud je splněna podmínka `<condition>`, provede se `BLOKA`

while cykly

FF UK
Department of Logic

```
while <condition>:
  BLOKA
BLOKB
```

- pokud je splněna podmínka `<condition>`, provede se `BLOKA`
- předchozí krok se opakuje dokud je splněna podmínka `<condition>`

while cykly

FF UK
Department of Logic

```
while <condition>:
  BLOKA
BLOKB
```

- pokud je splněna podmínka `<condition>`, provede se `BLOKA`
- předchozí krok se opakuje dokud je splněna podmínka `<condition>`
- pokud python narazí na příkaz `break`, ukončí pokračuje vykonáváním `BLOKB`

while cykly

FF UK
Department of Logic

```
while <condition>:
  BLOKA
BLOKB
```

- pokud je splněna podmínka `<condition>`, provede se `BLOKA`
- předchozí krok se opakuje dokud je splněna podmínka `<condition>`
- pokud python narazí na příkaz `break`, ukončí pokračuje vykonáváním `BLOKB`
- "zajišťují Turingovskou úplnost"

while cykly příklad

FF UK
Department of Logic

```
C = 0
M = 0
P = -1
S = 0

while C >= 0:
    S += C
    if C > M:
        M = C
    P += 1
    C = input('Cislo (zaporne ukonci):')

print("P:", P)
print("M:", M)
print("A:", S/P)
```

Funkce

FF UK
Department of Logic

```
def pozdrav( kdo ):
    print("Ahoj", kdo + "!")
```

```
>>> pozdrav("Petre")
Ahoj Petre!
```

```
def <jmeno>(<parametr_1>[=v1],...,<parametr_n>[=vn]):
    <telo_funkce>
```

Funkce

FF UK
Department of Logic

```
def pozdrav( kdo ):
    print("Ahoj", kdo + "!")
```

```
>>> pozdrav("Petre")
Ahoj Petre!
```

```
def <jmeno>(<parametr_1>[=v1],...,<parametr_n>[=vn]):
    <telo_funkce>
```

Jak dostat data z funkce?

Funkce

vracení hodnot

FF UK
Department of Logic

```
def quadsolve(a,b,c,x1,x2):
    D=b**2-4*a*c
    x1 = -1*b+sqrt(D)
    x2 = -1*b-sqrt(D)
```

```
>>> quadsolve(10,-100,1,x1,x2)
Traceback (most recent call last):
  File "<pyshell#17>", line 1, in <module>
    quadsolve(10,-100,1,x1,x2)
NameError: name 'x1' is not defined
```

```
>>> x1=0
>>> x2=0
>>> quadsolve(10,-100,1,x1,x2)
>>> print(x1, x2)
0 0
```

FF UK
Department of Logic

```
return
```

return

```
def quadsolve(a,b,c):
    D=b**2-4*a*c
    try:
        x1 = -1*b+sqrt(D)
        x2 = -1*b-sqrt(D)
        return x1, x2
    except:
        return None
```

```
>>> print quadsolve(10,10,10)
None
>>> print quadsolve(1,-10,10)
(17.745966692414832, 2.254033307585166)
```

Funkce

vracení hodnot

FF UK
Department of Logic

```
def quadsolve(a,b,c,vysledky):
    D=b**2-4*a*c
    if D >= 0:
        x1 = -1*b+sqrt(D)
        x2 = -1*b-sqrt(D)
        if D > 0:
            vysledky.append(x1)
            vysledky.append(x2)
        else:
            vysledky.append(x1)
>>> x=[]
>>> quadsolve(10,10,10,x)
>>> x
[]
>>> quadsolve(10,-100,10,x)
>>> x
```

Funkce

vracení hodnot

FF UK
Department of Logic

```
>>> def test(x):
        x=[]
>>> x
[0.0]
>>> test(x)
>>> x
[0.0]
```

Parametry Shrnutí

- parametry se předávají “hodnotou”, t.j.

```
>>> def test(x):
        x=[]
>>> x
[0.0]
>>> test(x)
>>> x
[0.0]
```

Parametry Shrnutí

- parametry se předávají "hodnotou", t.j.
- změnit lze pouze obsah "mutable" proměnných (seznamy, dict, některé další)