Podmínečné opakování

Jiří Zacpal

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

KMI/ZPP1 Základy programování v Pythonu 1

Řešení minulého úkolu

```
n = 100
for x in range(3, n + 1):
    prvocislo1=True
    prvocislo2=True
    for i in range(2, x):
        if x % i == 0:
            prvocislo1=False
        if (x + 2) % i == 0:
            prvocislo2=False
    if prvocislo1 and prvocislo2:
        print(x, x + 2)
```

Řešení minulého úkolu

```
n=100

for c in range(3,n+1):
    for i in range(2,c):
        if c%i==0 or (c+2)%i==0:
            break
    else:
        print(c,c+2)
```

Podmínečné opakování

Příklad

- Chceme vytisknout jednotlivé číslice tříciferného čísla:

```
n = 123
n1 = n

c = n1 % 10
n1 = n1 // 10
print(c)

c = n1 % 10
n1 = n1 // 10
print(c)

c = n1 % 10
n1 = n1 // 10
print(c)
```

Příklad

- Použijeme rozšířený příkaz přiřazení:

```
n = 123
n1 = n

c = n1 % 10
n1 //= 10
print(c)

c = n1 % 10
n1 //= 10
print(c)

c = n1 % 10
n1 //= 10
print(c)
```

Příklad

- Použijeme cyklus for:

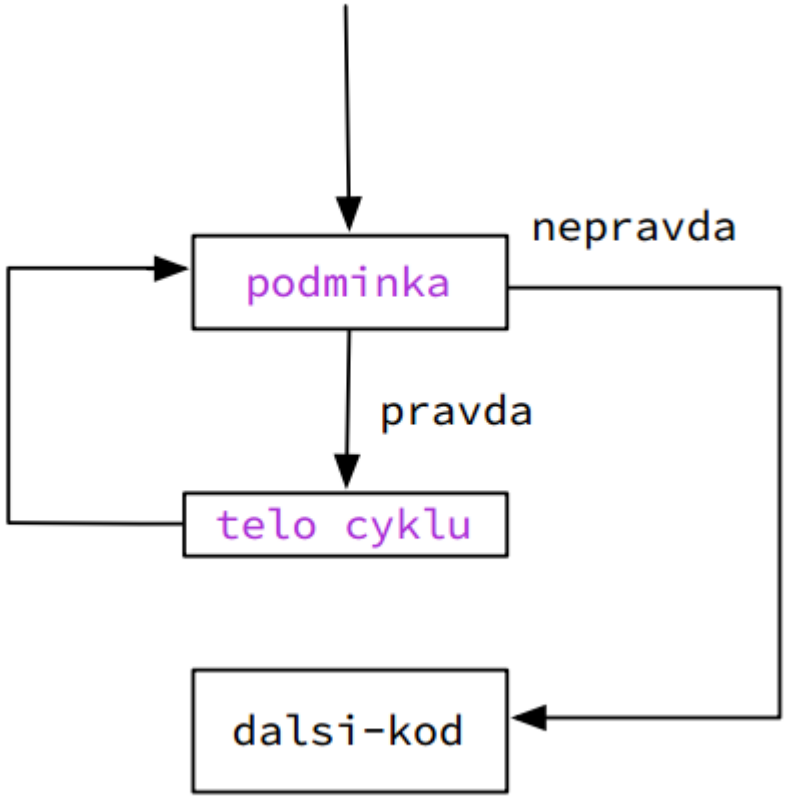
```
n = 123
n1 = n
for i in range(3):
    c = n1 % 10
    n1 //= 10
    print(c)
```

- Co ale když chceme vytisknout jednotlivé číslice obecně n-ciferného čísla a předem nevíme, kolik je n?

Cyklus while

- syntaxe:

```
while (podminka):
    telo cyklu
```



Příklad

- Použijeme cyklus while na předchozí úkol:

```
n = 123
n1 = n

while n1 != 0:
    c = n1 % 10
    n1 //= 10
    print(c)
```

Nekonečný cyklus

- Cyklus, který nikdy neskončí:

```
while True:
    print(0)
```

- Ne vždy je pád do nekonečné smyčky takto průzračný:

```
n = 10
n1 = n
while n1 != 0:
    print(n1)
    n1 -= 2
```

- Co ale když n bude 9?
- Chyba je v podmínce. Jak ji upravíme?

```
while n1 >= 0:
```

Kontrola platnosti vstupu (nezáporné číslo)

- Někdy však zdánlivě nekonečný cyklus požadujeme:

```
a=int(input("Zadej číslo: "))
while (a < 0):
  a=int(input("Zadej číslo znovu: "))
print(f"Zadal jste číslo: {a}")
```

Přepis cyklu for pomocí while

- Vezměme zápis cyklu for:

```
for i in range(v):
    příkazy
```

- Můžeme přepsat pomocí while:

```
n = v
i = 0
while i < n:
    příkazy
    i += 1
```

- Cyklus while nemůžeme obecně přepsat na cyklus for a to z toho důvodu, že cyklus for narozdíl od cyklu while vždy skončí.

Příklad

- Vraťme se k určování prvočísla:

```
n = 7
je_prvocislo = True
for i in range(2,n):
    if n % i == 0:
        je_prvocislo= False
print(je_prvocislo)
```

- Cyklus celkem zbytečně probíhá, i když už je je_prvočíslo False. Můžeme použít break, ale lepší je použít cyklus while:

```
n = 7
je_prvocislo = True
i=2

while i<n and je_prvocislo==True:
    if n % i == 0:
        je_prvocislo= False
    i+=1

print(je_prvocislo)
```

Přerušení iterace

Příklad

- Někdy by bylo výhodné přerušit iteraci způsobenou příkazem cyklu for:

```
n = 7

je_prvocislo = True
for i in range(2,n):
    if n % i == 0:
        je_prvocislo= False

print(je_prvocislo)
```

Příkazy přerušení cyklu

- příkaz `continue`
  - skok na konec nejvnitřnějšího cyklu, výpočet pokračuje další iterací (včetně testu případné podmínky)
- příkaz `break`
  - okamžité opuštění nejvnitřnějšího cyklu
- Vztahují se vždy k „nejbližšímu“ cyklu

Příklad

- Někdy by bylo výhodné přerušit iteraci způsobenou příkazem cyklu for:

```
n = 7

je_prvocislo = True
for i in range(2,n):
    if n % i == 0:
        je_prvocislo= False
        break

print(je_prvocislo)
```

- Příkaz break se musí nacházet v těle cyklu:

```
>>> break
  File "<stdin>", line 1
SyntaxError: 'break' outside loop
>>>
```

Příklad

- Vezměme si nyní následující program, který tiskne dvojice nezáporných čísel menších než zadané číslo:

```
n = 10
for i in range(n):
    for j in range(n):
        print(i, j)
```

- Co když budeme chtít program upravit tak, aby první číslo bylo menší nebo rovno než druhé číslo:

```
n = 10
for i in range(n):
    for j in range(n):
        if i<=j:
            print(i, j)
```

- Není to efektivní.

Příklad

- Program upravíme takto:

```
n = 10
for j in range(n):
    for i in range(n):
        if j < i:
            break
        print(i, j)
```

- Dá se program napsat bez break?

```
n = 10
for j in range(n):
    for i in range(j + 1):
        print(i, j)
```

Příklad

- Nalezení x-tého čísla dělitelného číslem n, které je větší nebo rovno číslu od. Navíc vypisujeme všechna testovaná čísla, která nejsou dělitelná číslem n.

```
n=int(input("Zadej číslo n: "))
x=int(input("Zadej číslo x: "))
od=int(input("Zadej číslo od: "))

nalezeno=0
i=od-1
while (True):
    i+=1
    if (i%n==0):
        nalezeno+=1
        if (nalezeno==x):
            break
        else:
            continue
    print(f"{i},",end=" ")

print(f"\n{x}-té číslo dělitelné {n} od čísla {od} je číslo {i}\n")
```

Cyklus s podmínkou na konci

Cyklus s podmínkou na konci

- Python nenabízí, jako jiné jazyky, cyklus s podmínkou na konci = cyklus, který alespoň jednou proběhne.
- Dá se tento cyklu vytvořit pomocí klasického cyklu `while` a příkazu `break`.

- **Příklad:**
- Požadujeme, aby uživatel zadal správné heslo.

```
while True:
    heslo=input("Zadej heslo:")
    if heslo=="Python":
        break
    else:
        print("Špatné heslo.")
```