Podmienky, príkaz if, cyklus while a break

1 Podmienka a typ bool

Podmienka je výraz, ktorého pravdivosť rozhoduje o vykonaní príkazu alebo opakovaní cyklu. Výsledkom porovnania je objekt typu `bool` s hodnotou `True` alebo `False`.

Porovnávacie operátory sú `<`, `<=`, `>`, `>=`, `==` a `!=`. Operátor `=` priraďuje hodnotu; operátor `==` porovnáva rovnosť. Zápis `a != b` znamená, že hodnoty nie sú rovnaké.

```
a = 7
b = 10
print(a < b, a >= b)
print(a == 7, a != b)
print(type(a < b))
```

Číselné typy možno pri porovnaní bežne kombinovať. Reťazce sa porovnávajú lexikograficky podľa kódov Unicode; záleží na veľkosti písmen a diakritike. Nepríbuzné typy sa operátormi usporiadania zvyčajne porovnať nedajú.

Python umožňuje reťazenie porovnaní. Zápis `dolna <= x <= horna` znamená to isté ako `dolna <= x and x <= horna`.

```
x = 6
print(0 <= x <= 10)
print(x < 0 or x > 10)
print(not (x == 6))
```

V podmienke sa na pravdivostnú hodnotu prevedú aj iné objekty. Čísla `0` alebo `0.0` a prázdny reťazec sú nepravdivé `False`. Nenulové čísla a neprázdne objekty sú spravidla pravdivé. Funkcia `bool()` ukáže výsledok prevodu.

2 Logické operátory

Operátor `and` vyžaduje pravdivosť oboch podmienok. Operátor `or` vyžaduje pravdivosť aspoň jednej. Operátor `not` pravdivosť obráti. Priorita je: porovnania, `not`, `and`, `or`; zátvorky význam spresnia. Operátory `and` a `or` používajú skrátené vyhodnocovanie. Pri `A and B` sa výraz `B` nevyhodnotí, ak je `A` nepravdivé. Pri `A or B` sa `B` nevyhodnotí, ak je `A` pravdivé.

```
x = float(input("Zadaj x: "))
y = float(input("Zadaj y: "))
sirka, vyska = 12.0, 8.0
if 0 <= x <= sirka and 0 <= y <= vyska:
    print("Bod leží v obdĺžniku alebo na jeho hranici.")
else:
    print("Bod leží mimo obdĺžnika.")
```

3 Príkazy if, elif a else

`if podmienka:` vykoná odsadený blok iba pri pravdivej podmienke. Vetva `else` je voliteľná a vykoná sa pri nepravdivej podmienke. Príkaz `pass` môže dočasne zastúpiť prázdny blok.

Medzi `if` a `else` môže byť ľubovoľný počet vetiev `elif`. Podmienky sa skúšajú zhora nadol a vykoná sa iba prvá pravdivá vetva. Posledný výraz `else` zachytí všetky zostávajúce prípady.

```
cislo = int(input("Zadaj celé číslo: "))
if cislo > 0:
    print("kladné číslo")
elif cislo < 0:
    print("záporné číslo")
else:
    print("nula")
```

Vetvy patria k sebe iba pri rovnakom odsadení. Vnorený `if` vytvára ďalšie rozhodnutie vo vnútri vetvy.

4 Prevod bodov na známku

```
body = int(input('Zadaj získaný počet bodov: '))
if body >= 90:
    print('za', body, 'bodov získavaš známku A')
elif body >= 80:
    print('za', body, 'bodov získavaš známku B')
elif body >= 70:
    print('za', body, 'bodov získavaš známku C')
elif body >= 60:
    print('za', body, 'bodov získavaš známku D')
elif body >= 50:
    print('za', body, 'bodov získavaš známku E')
else:
    print('za', body, 'bodov si nevyhovel a máš známku Fx')
```

Hranice sú usporiadané zostupne. Hodnota `95` spĺňa aj nižšie hranice, ale vykoná sa prvá pravdivá vetva.

Program predpokladá platný celočíselný vstup. Ak majú byť povolené iba body od `0` do `100`, treba rozsah osobitne overiť podmienkou `0 <= body <= 100`.

5 Delitele čísla

Zvyšok po delení vypočíta operátor `%`. Číslo `delitel` je deliteľom čísla `cislo`, ak platí `cislo % delitel == 0`.

```
cislo = int(input("Zadaj kladné číslo: "))
pocet = 0
print("Delitele:", end=" ")
for delitel in range(1, cislo + 1):
    if cislo % delitel == 0:
        pocet += 1
        print(delitel, end=" ")
print()
print("Počet deliteľov:", pocet)
```

Cyklus skúša všetky celé čísla od `1` po `cislo`. Počítadlo sa zväčší a deliteľ sa vypíše iba vtedy, keď je zvyšok nulový.

6 Cyklus while

Cyklus `while podmienka:` opakuje odsadený blok, kým je podmienka pravdivá. Podmienka sa testuje pred každým opakovaním, preto sa telo nemusí vykonať ani raz. Riadiaca premenná sa zvyčajne inicializuje pred cyklom a mení v jeho tele. Ak sa stav podmienky nemôže zmeniť na `False`, vznikne nekonečný cyklus.

```
i = 1
while i <= 5:
    print(i)
    i += 1
print("koniec")
```

Príkaz `for` sa používa pri prechádzaní známej postupnosti. Príkaz `while` sa používa, keď počet opakovaní vopred nepoznáme a pokračovanie závisí od stavu alebo vstupu. Často sa používa, ak inkrement je reálne číslo.

```
sucet = 0
cislo = int(input("Zadaj číslo, nula končí: "))
while cislo != 0:
    sucet += cislo
    cislo = int(input("Zadaj číslo, nula končí: "))
print("Súčet =", sucet)
```

Nula je ukončovacia hodnota nazývaná zarážka. Do súčtu sa nepridá, pretože podmienka sa overí pred vstupom do tela príkazu `while`.

7 Príkazy break, continue a else pri cykle

Príkaz `break` okamžite ukončí najbližší cyklus `for` alebo `while`. Vnorené cykly neukončí všetky, iba cyklus, v ktorom sa priamo nachádza.

```
while True:
    cislo = int(input("Zadaj kladné číslo: "))
    if cislo > 0:
        break
    print("Číslo musí byť kladné.")
print("Prijatá hodnota:", cislo)
```

Podmienka `True` vytvára nekonečný cyklus. Platný vstup vykoná `break`; neplatný vstup pokračuje ďalším opakovaním.

Príkaz `continue` preskočí zvyšok aktuálneho opakovania a začne ďalšie. Pri `while` treba zabezpečiť, aby sa pred `continue` správne zmenil stav riadiaci podmienku.

```
for cislo in range(1, 11):
    if cislo % 2 == 0:
        continue
    print(cislo, end=" ")
```

Voliteľná vetva `else` cyklu `for` sa vykoná po normálnom skončení cyklu. Nevykoná sa, ak cyklus ukončil `break`.

8 Test prvočísla

Nasledujúci postup skúša delitele od 2. Prvý nájdený deliteľ ukončí cyklus príkazom `break`. Vstup musí byť celé číslo aspoň 2.

```
cislo = int(input("Zadaj číslo aspoň 2: "))
for delitel in range(2, cislo + 1):
    if cislo % delitel == 0:
        break
if delitel == cislo:
    print(cislo, "je prvočíslo")
else:
    print(cislo, "nie je prvočíslo")
```

Každé číslo je deliteľné samým sebou. Ak prvý nájdený deliteľ je priamo `cislo`, menší netriviálny deliteľ neexistuje. Ak sa nájde menší deliteľ, číslo je zložené.

Rovnaký algoritmus možno zapísať cyklom `while`. Podmienka spája dve požiadavky: deliteľ je ešte menší ako číslo a zatiaľ nedelí bez zvyšku.

```
cislo = int(input("Zadaj číslo aspoň 2: "))
delitel = 2
while delitel < cislo and cislo % delitel != 0:
    delitel += 1
if delitel == cislo:
    print(cislo, "je prvočíslo")
else:
    print(cislo, "nie je prvočíslo")
```

Skúšať delitele až po samotné číslo je zbytočné. Jednoduché zrýchlenie použije `range(2, cislo // 2 + 1)`. Stačí však skúšať iba po odmocninu, pretože pri rozklade `a·b = cislo` nemôžu byť oba činitele väčšie ako odmocnina.

```
import math

cislo = int(input("Zadaj číslo: "))
if cislo < 2:
    print(cislo, "nie je prvočíslo")
else:
    for delitel in range(2, int(math.sqrt(cislo)) + 1):
        if cislo % delitel == 0:
            print(cislo, "nie je prvočíslo")
            break
    else:
        print(cislo, "je prvočíslo")
```

Výraz `else` patrí k cyklu `for`: vykoná sa iba vtedy, keď sa cyklus skončí bez príkazu `break`.