25.

Podmínky a cykly: while, for, if v různých jazycích, pass, continue, break

Podmínky (if):

Podmínka je konstrukce v programování, která umožňuje rozhodování o tom, který blok kódu se má provést na základě splnění určitého logického výrazu.

Syntaxe:

```
if podmínka:
    provedi_kód
elif jiná_podmínka:    # volitelné
    provedi_kód
else:                   # volitelné
    provedi_kód
```

podmínka je libovolný výraz, který se vyhodnotí na pravdivý (True) nebo nepravdivý (False).
provedi_kód je blok kódu, který se provede, pokud je podmínka splněna.
elif (zkratka pro "else if") umožňuje definovat další podmínky ke kontrole.
else definuje blok kódu, který se provede, pokud žádná předchozí podmínka není splněna.

Cyklus while:

Cyklus while opakuje blok kódu, dokud je jeho podmínka pravdivá.

Syntaxe:

```
while podmínka:
    provedi_kód
```

podmínka je vyhodnocována před každou iterací cyklu.
Pokud je podmínka pravdivá, provede se blok kódu. Poté se podmínka znovu vyhodnotí.
Cyklus while může vést k nekonečné smyčce, pokud se podmínka nikdy nestane nepravdivou.

Cyklus for:

Cyklus for slouží k opakování určitého bloku kódu pro každý prvek v sekvenci (například seznamu nebo řetězci).

Syntaxe:

```
for prvek in sekvence:
    provedi_kód
```

prvek je proměnná, která přebírá hodnoty z sekvence v každé iteraci.
sekvence může být seznam, n-tice, řetězec nebo jiná iterovatelná struktura.
Cyklus for automaticky projde všechny prvky sekvence a provede blok kódu pro každý z nich.

Pass

V Pythonu je pass nulová operace
Nic se nestane, když je provedena.
Obvykle se používá jako zástupný prvek, když syntax vyžaduje příkaz, ale nemáte žádnou akci k provedení.

Syntaxe:

```
if x < 0:
    pass  # Pro záporné hodnoty x nic nedělejte
else:
    print("x je kladné")
```

Continue

Příkaz "continue" se používá uvnitř smyček k přeskočení zbývajícího kódu uvnitř smyčky pro aktuální opakování a pokračování na další opakování. Například:,

Syntaxe:

```
for i in range(5):
    if i == 2:
        continue  # Přeskočit opakování pro i = 2
    print(i)
```

Break

Předčasné ukončení smyčky, bez ohledu na podmínku smyčky nebo počet zbývajících iterací. Často se používá, když je splněna určitá podmínka, a chcete zastavit provádění smyčky.

Syntaxe:

```
for i in range(5):
    if i == 3:
        break  # Opusťte smyčku, když i = 3
    print(i)
```