PYTHON

Adam Klepáč

7. listopadu 2022

Gymnázium Evolution Jižní Město

Programovací jazyky

Nejnižší forma komunikace

Strojový kód

Strojový kód je jazyk sestávající pouze ze základních instrukcí pro CPU.

Nejvyšší forma komunikace

Programovací jazyk

Programovací jazyk je jakýkoli jazyk, který lze automaticky přeložit do strojového kódu.

Nač programovací jazyky?

- Strojový kód je člověku nečitelný.

Nač programovací jazyky?

- Strojový kód je člověku nečitelný.
- Programovací jazyky se čím dál více přibližují lidské řeči.

Nač programovací jazyky?

- Strojový kód je člověku nečitelný.
- Programovací jazyky se čím dál více přibližují lidské řeči.
- Programovací jazyky jsou rozšiřitelné – umožňují přidání nových konceptů (proměnných, podmínek, ...)

Nač programovací jazyky?

- Strojový kód je člověku nečitelný.
- Programovací jazyky se čím dál více přibližují lidské řeči.
- Programovací jazyky jsou rozšířitelné – umožňují přidání nových konceptů (proměnných, podmínek, ...)
- V programovacích jazycích lze některé běžné paměťové operace CPU automatizovat (rekurze, garbage collector, ...)

Jaká je cena?

- Překlad prog. jazyků je automatický – vzniká spousta přebytečného strojového kódu.

Jaká je cena?

- Překlad prog. jazyků je automatický – vzniká spousta přebytečného strojového kódu.
- Přebytečné instrukce zpomalují běh programu.

Jaká je cena?

- Překlad prog. jazyků je automatický – vzniká spousta přebytečného strojového kódu.
- Přebytečné instrukce zpomalují běh programu.
- Velká práce s údržbou – každá nová funkce programovací jazyka vyžaduje mnoho testování správnosti překladu do stroj. kódu

Jaká je cena?

- Překlad prog. jazyků je automatický – vzniká spousta přebytečného strojového kódu.
- Přebytečné instrukce zpomalují běh programu.
- Velká práce s údržbou – každá nová funkce programovací jazyka vyžaduje mnoho testování správnosti překladu do stroj. kódu
- V různých jazycích jsou stejné funkce psané jinak.

Typy programovací jazyků

(1) strojový kód,

Typy programovací jazyků

(1) strojový kód,

(2) assemblery (jazyky symbolických adres):

- symbolické reprezentace CPU instrukcí
- zkratky pro běžné operace
- žádná automatizace

Typy programovací jazyků

(1) strojový kód,

(2) assemblery (jazyky symbolických adres):

- symbolické reprezentace CPU instrukcí
- zkratky pro běžné operace
- žádná automatizace

(3) high-level (vysokoúrovňové) programovací jazyky:

- pokročilé řídící sekvence – proměnné, podmínky, cykly, ...
- automatická správa běhu – procedury, funkce
- částečně automatická správa paměti – pole, třídy, ...

Kam patří Python?

- Python je high-level programovací jazyk.

Kam patří Python?

- Python je high-level programovací jazyk.
- Python $\rightarrow$ C $\rightarrow$ (Assembly $\rightarrow$) stroj. kód

Kam patří Python?

- Python je high-level programovací jazyk.
- Python $\rightarrow$ C $\rightarrow$ (Assembly $\rightarrow$) stroj. kód
- Python je interpretovaný (vs. kompilovaný) programovací jazyk – to znamená, že počítač překládá Python za běhu programu.

Kam patří Python?

- Python je high-level programovací jazyk.
- Python $\rightarrow$ C $\rightarrow$ (Assembly $\rightarrow$) stroj. kód
- Python je interpretovaný (vs. kompilovaný) programovací jazyk – to znamená, že počítač překládá Python za běhu programu.
- Python má automatickou správu paměti a dokonce vás ani nenutí typovat.

I. Programování v Pythonu

Datové typy

Co to je?

Datový typ

Datový typ je doslova typ (forma, podoba, ...) informace uložené v paměti počítače.

Co to je?

Datový typ

Datový typ je doslova typ (forma, podoba, ...) informace uložené v paměti počítače.

- Narozdíl od pseudokódu, v programovacích jazycích musíte kromě názvu proměnné uvádět i její typ.

Co to je?

Datový typ

Datový typ je doslova typ (forma, podoba, ...) informace uložené v paměti počítače.

- Narozdíl od pseudokódu, v programovacích jazycích musíte kromě názvu proměnné uvádět i její typ.
- Základní typy v Pythonu jsou `int`, `float`, `str`, `set`, `list`, `tuple`, `dict`

Měnné vs. neměnné

- Python rozlišuje mezi měnnými (mutable) a neměnnými (immutable) datovými typy.

Měnné vs. neměnné

- Python rozlišuje mezi měnnými (mutable) a neměnnými (immutable) datovými typy.
- Do struktury měnných typů (seznamy, slovníky, ...) můžete zasahovat během programu, ale do struktury neměnných (čísla, slova, ...) nikoliv.

Datové typy

Číselné typy

Celá čísla

Datový typ `int`

Zkratkou `int` (z angl. integer) Python označuje typ celých čísel, tj. čísel bez desetinné části.

Celá čísla

Python umí následující operace na celých číslech.

- součet (+);
- rozdíl (−);
- součin (∗);

Celá čísla

Python umí následující operace na celých číslech.

- součet (+);
- rozdíl (−);
- součin (∗);
- celočíselný podíl (//), např. `11 // 3 == 3`;

Celá čísla

Python umí následující operace na celých číslech.

- součet (+);
- rozdíl (−);
- součin (∗);
- celočíselný podíl (//), např. `11 // 3 == 3`;
- zbytek po dělení (%), např. `11 % 3 == 2`;

Celá čísla

Python umí následující operace na celých číslech.

- součet (+);
- rozdíl (−);
- součin (∗);
- celočíselný podíl (//), např. `11 // 3 == 3`;
- zbytek po dělení (%), např. `11 % 3 == 2`;
- mocninu (∗∗), např. `4 ** 3 == 64`.

Desetinná čísla

Datový typ `float`

Zkratka `float` (z angl. floating point) označuje v Pythonu typ desetinných čísel.

Desetinná čísla

Datový typ `float`

Zkratka `float` (z angl. floating point) označuje v Pythonu typ desetinných čísel.

Poznámka. Celá čísla jsou samozřejmě zároveň desetinná. Aby je Python v tomto případě rozlišil, píše `2.0` pro „desetinné číslo“ dva a `2` pro „celé číslo“ dva.

Desetinná čísla

Python umí následující operace na desetinných číslech.

- součet (+);
- rozdíl (−);
- součin (*);
- podíl (/);
- mocninu (**).

celá ↔ desetinná

- Slova `int` a `float` jsou zároveň názvy funkcí/procedur v Pythonu pro převod mezi číselnými typy.

celá ↔ desetinná

- Slova `int` a `float` jsou zároveň názvy funkcí/procedur v Pythonu pro převod mezi číselnými typy.
- `int(x: float)` vrátí tzv. „celou část“ z `x`; např. `int(3.9) == 3`.

celá ↔ desetinná

- Slova `int` a `float` jsou zároveň názvy funkcí/procedur v Pythonu pro převod mezi číselnými typy.
- `int(x: float)` vrátí tzv. „celou část“ z `x`; např. `int(3.9) == 3`.
- `float(x: int)` převede celé číslo `x` na desetinné prostě tak, že k němu přidá „`.0`“. Takže třeba `float(3) == 3.0`.

Řetězce

Řetězce (stringy)

Datový typ `str`

Zkratkou `str` (z angl. string) Python označuje typ „řetězců znaků“, tj. posloupností v zásadě libovolných symbolů.

Řetězce (stringy)

Datový typ `str`

Zkratkou `str` (z angl. string) Python označuje typ „řetězců znaků“, tj. posloupností v zásadě libovolných symbolů.

- Stringy se píší do uvozovek, buď jednoduchých (`'`) nebo dvojitých (`"`). Na výběru nezáleží, ale string musí začínat končit stejnou uvozovkou.

Řetězce (stringy)

Datový typ `str`

Zkratkou `str` (z angl. string) Python označuje typ „řetězců znaků“, tj. posloupností v zásadě libovolných symbolů.

- Stringy se píší do uvozovek, buď jednoduchých (`'`) nebo dvojitých (`"`). Na výběru nezáleží, ale string musí začínat končit stejnou uvozovkou.
- Python používá pro kódování textu UTF-8 (**U**nicode **T**ransformation **F**ormat – 8-bit). Tedy umí rozpoznat každý znak v tomto kódování.

Řetězce (stringy)

Python umí následující operace na řetězcích.

- součet/spojení (+ nebo mezera)
  - např. `"auto" + "bus" == "autobus"`
  - např. `"mrt" "vola" == "mrtvola"`

Řetězce (stringy)

Python umí následující operace na řetězcích.

- součet/spojení (+ nebo mezera)
  - např. `"auto" + "bus" == "autobus"`
  - např. `"mrt" "vola" == "mrtvola"`
- součin/opakování (*): např. `"hehe" * 3 == "hehehehehehe"`

Řetězce (stringy)

Python umí následující operace na řetězcích.

- součet/spojení (+ nebo mezera)
  - např. `"auto" + "bus" == "autobus"`
  - např. `"mrt" "vola" == "mrtvola"`
- součin/opakování (`*`): např. `"hehe" * 3 == "hehehehehehe"`
- výběr prvku(`[pořadí prvku]`): např. `"python"[2] == "t"`.
  Pozor! Python čísluje od 0.

stringy ↔ čísla

- Zkratka `str` je zároveň procedura na převod dané proměnné na string. V případě čísel máme
  - `str(x: int)` převede celé číslo na string. Třeba `str(3) == "3"`.
  - `str(x: float)` převede desetinné číslo na string. Např. `str(3.14159) == "3.14159"`.

  Pozor! Python neřeší, jestli je ve stringu číslo. Takže třeba `"1" + "1" == "11"`, ale `1 + 1 == 2`.

stringy ↔ čísla

Procedury `int` a `float` taky převádějí stringy na číslo, pokud to lze. Např.

- `int("69") == 69`,

stringy ↔ čísla

Procedury `int` a `float` taky převádějí stringy na číslo, pokud to lze. Např.

- `int("69") == 69`,
- `float("3.14159") == 3.14159`, ale

stringy ↔ čísla

Procedury `int` a `float` taky převádějí stringy na číslo, pokud to lze. Např.

- `int("69") == 69`,
- `float("3.14159") == 3.14159`, ale
- `float("hehe")` i `int("9.11")` hodí chybu.

Díky za pozornost.