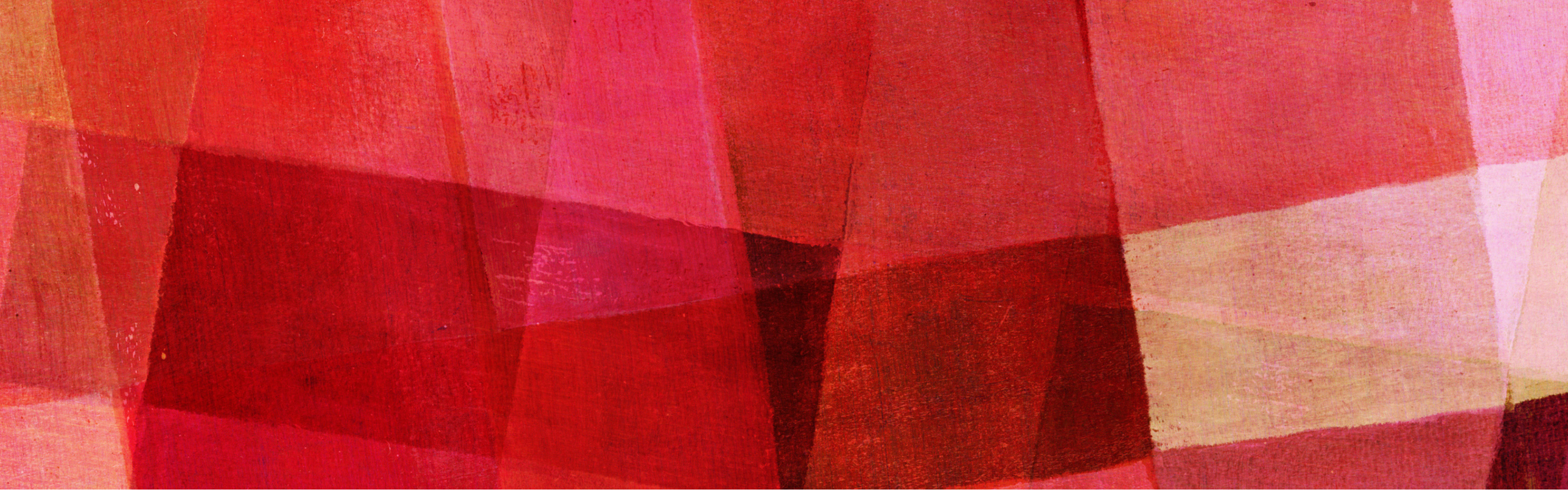

Programování 1 (NPRG030)

Cvičení 1: Úvod

Požadavky na zisk zápočtu

Cvičení

- Účast na cvičení *není povinná*, ale je silně *doporučována*
- Na každém cvičení lze za *aktivní řešení příkladů* získat až 2 *body*

Domácí úkoly

- Během semestru Vám bude zadáno *10 domácích úkolů*
- Řešení domácího úkolu se *odevzdává do ReCoDeXu* a odevzdávat *lze opakovaně*
- Na *vyřešení* domácího úkolu budete mít vždy *alespoň 13 dnů*
- Za každý *vyřešený* domácí úkol lze získat *nejvýše 10 bodů*
- *Pozdě odevzdané* domácí úkoly budou *penalizovány -2 body za každý započatý týden* zpoždění

Semestrální práce

- Během semestru vypracujete *zdokumentované a otestované softwarové dílo*
  - Konkrétní požadavky jsou uvedeny v samostatném PDF souboru (bude k dispozici do konce čtvrtého týdne)
- Za semestrální práci lze získat *maximálně 40 bodů*. Každý *nesplněný požadavek* bude *penalizován*
- Pokud nezískáte alespoň 30 bodů za semestrální práci, máte ještě jeden *opravný pokus*. V případě opravy *musíte získat 35 bodů*
- Pokud nezískáte alespoň 30 bodů na první pokus nebo alespoň 35 bodů z opravy, ztrácíte nárok na udělení zápočtu

Požadavky na zisk zápočtu

Zápočtový test

- Na konci semestru proběhne *praktický programovací test*
  - Zadávaný a odevzdávaný do ReCoDeXu
- Čas na vyřešení bude maximálně *90 minut*
- Test je třeba splnit na *100% bodů*
- Na test budete mít *tři pokusy*

Pro udělení zápočtu musíte splnit následující

- Odevzdat *všech 10 domácích úkolů* a získat *alespoň 60 bodů* za řešení domácích úkolů a aktivitu během cvičení
- Úspěšně *odevzdat semestrální práci*
- Úspěšně *projít zápočtovým testem*

Programovací prostředí (IDE)

Visual Studio Code

- Editor zdrojového kódu pro operační systémy Linux, macOS a Windows
- Obsahuje podporu pro Git, zvýraznění syntaxe a kontextový našeptávač
- Odkaz pro stažení: https://code.visualstudio.com/
- Pro podporu Pythonu je nutná instalace rozšíření Python
  - https://marketplace.visualstudio.com/items?itemName=ms-python.python

PyCharm

- IDE pro Python pro operační systémy Linux, macOS a Windows
- Odkaz pro stažení: https://www.jetbrains.com/pycharm/

Apache NetBeans IDE

- IDE původně pro Javu pro operační systémy Linux, macOS a Windows
- Odkaz pro stažení: https://netbeans.apache.org/
- Pro podporu Pythonu je nutná instalace rozšíření Python

Shrnutí přednášky: Základy

```
# Proměnné a konstanty
jmeno = "Tomáš" # Přiřadí řetězec 'Tomáš' do proměnné `jmeno`
jmeno: str = "Adéla" # Deklaruje proměnnou `jmeno` jako řetězec a přiřadí jí hodnotu 'Adéla'
JMENO_UNIVERZITY = "Univerzita Karlova" # Vytvoří konstantu (pouze na základě jmenné konvence)
```

```
# Převádění typů
bool(object) # Převede prázdný řetězec nebo 0 na False a jiné hodnoty na True
int(object) # Převede řetězec nebo ořízne reálné číslo na celé číslo (může vrátit ValueError)
float(object) # Převede řetězec na desetinné číslo (může dojít k ValueError)
str(object) # Převede libovolnou hodnotu na řetězec
```

```
# Vstup a výstup
input() # Získá uživatelův vstup z klávesnice
print(object1, object2) # Vypíše na řádek hodnoty
print(object1, object2, sep=oddelovac) # Vypíše na řádek hodnoty oddělené oddělovačem
print(object1, object2, end=endline) # Vypsané hodnoty jsou ukončeny pomocí endline (např. \n)
```

```
# Užitečné funkce
type(object) # Vrátí typ objektu
abs(hodnota) # Vrátí absolutní hodnotu čísla
min(hodnota1, hodnota2) # Vrátí menší hodnotu ze dvou
max(hodnota1, hodnota2) # Vrátí větší hodnotu ze dvou
ord(znak) # Vrátí celočíselnou hodnotu Unicode znaku
chr(hodnota) # Vrátí znak na základě Unicode hodnoty
```

Shrnutí přednášky: Základy II

```
# Toto je komentář na jeden řádek

"""
Toto je komentář
na více řádků
"""
```

```
# Vybrané operátory
+  # Sčítání
-  # Odčítání
*  # Násobení
// # Celočíselné dělení
%  # Zbytek po dělení
/  # Dělení s výsledkem typu float
** # Umocňování
==, !=, <, >, <=, >=  # Porovnávání dvou hodnot, objektů nebo výrazů
```

```
# Podmíněný příkaz
if cislo1 > cislo2:                # Testovaný výraz
  print("První číslo je větší") # Operace se provede, pokud je výraz vyhodnocen na True
elif cislo2 > cislo1:              # Testovaný výraz druhý v pořadí
  print("Druhé číslo je větší") # Operace se provede, pokud druhý testovaný výraz je pravdivý
else:
  print("Čísla jsou stejná")    # Vykoná se, pokud ani jeden z výrazů není pravdivý
```

Příklad 1.4: Hello world!

- Vytvořte *program* v jazyce Python, který Vás po zadání jména *pozdraví*

Cvičení 1.5: Triviální kalkulačka

- Vytvořte *program* v jazyce Python, který *pro* uživatelem *zadané* celé *číslo vypočítá*:
  - *Součet* zadané hodnoty a čísla 7
  - *Rozdíl* zadané hodnoty a čísla 7
  - *Násobek* zadané hodnoty a čísla 7
  - Výsledek (celočíselného) *dělení* zadané hodnoty a čísla 7
  - *Zbytek po dělení* zadané hodnoty a čísla 7
- Na výstup *vypište celé operace*, tj. včetně operandů, operátoru a výsledku
  - Jednotlivé operace vypište na nový řádek

Cvičení 1.6: Lepší kalkulačka

- Vylepšete kalkulačku z předchozího příkladu tak, aby pro celá i desetinná čísla umožňovala:
  - *Sečíst* dvě zadané hodnoty
  - *Odečíst* od sebe dvě hodnoty v zadaném pořadí
  - *Vynásobit* dvě hodnoty
  - *Vydělit* dvě hodnoty v zadaném pořadí

- Umožněte, aby uživatel zadal *na vstupu obě hodnoty*, a určil *operaci*, která se má vykonat
- Na výstup *vypište celou operaci*, tj. včetně operandů, operátoru a výsledku

- **Těžší varianta**
  - Co se stane, když uživatel zadá *jinou hodnotu než číslo* nebo *neplatný operátor*?
    - Zkuste *ošetřit zadání neplatného vstupu*

Jak psát kód: Komentáře a dokumentace

- Komentáře a dokumentace
  - Pište *jasné komentáře a dokumentaci*, abyste *vysvětlili účel* a chování kódu

Jak psát kód: Popisné názvy proměnných

- Zvolte a používejte *popisné názvy proměnných*, které vyjadřují účel proměnné

```
1 # Nejasný význam
2 a = 22
3 b = 10
```

```
1 # Jasný význam
2 day = 22
3 month = 10
```

Jak psát kód: Nepoužívejte "magic numbers"

- Používejte *pojmenované konstanty* namísto natvrdo zadaných čísel, abyste zvýšili čitelnost kódu

```
1 # Magic number
2 if age > 42:
3   print("Older than 42")
```

```
1 # Pojmenovaná konstanta
2 THRESHOLD = 42
3 if age > THRESHOLD:
4   print("Older than", THRESHOLD)
```

Jak psát kód: Keep It Simple, Stupid (KISS)

- Snažte se o *jednoduchost kódu*
  - Vyhněte se *zbytečné složitosti*, která může vést ke zmatení

```
# Složitá verze
if (month > 0 and month < 10) or (day > 10 and day < 20):
  print("Condition met")
```

```
# Jednodušší verze
if 0 < month < 10 or 10 < day < 20:
  print("Condition met")
```

Jak psát kód: Optimalizace kódu pro čitelnost

- Upřednostněte *čitelnost kódu* před chytrými optimalizacemi, které zastírají účel kódu

```
1 # Chytré, ale hůř čitelné
2 value = a and b or c
```

```
1 # Čitelná verze
2 if a:
3   value = b
4 else:
5   value = c
```

Jak psát kód: Trojčlenné výrazy

- Vyhněte se příliš složitým ternárním výrazům (angl. *ternary expression*) pro lepší čitelnost kódu

```
# Komplexní výraz
result = "Yes" if (tax > 0 and income < 10000) else "No"
```

```
# Rozdělení na if-else
if tax > 0 and income < 10000:
  result = "Yes"
else:
  result = "No"
```

Reference

Programovací prostředí (IDE)

- Visual Studio Code: https://code.visualstudio.com/
- PyCharm: https://www.jetbrains.com/pycharm/
- Apache NetBeans: https://netbeans.apache.org/

Syntaxe jazyku Python 3

- https://www.w3schools.com/python/python_syntax.asp
- https://www.w3schools.com/python/python_variables.asp
- https://www.w3schools.com/python/python_datatypes.asp
- https://www.w3schools.com/python/python_numbers.asp
- https://www.w3schools.com/python/python_casting.asp
- https://www.w3schools.com/python/python_operators.asp
- https://www.w3schools.com/python/python_comments.asp
- https://www.w3schools.com/python/python_user_input.asp
- https://www.w3schools.com/python/python_conditions.asp