**1. Názorně ukažte práci s přechody snímků v prezentaci, závěrečnou úpravu prezentace a způsoby uložení.**

**Práce s přechody snímků v prezentaci**

1. **Vytvoření prezentace:**
   * Otevřete aplikaci pro tvorbu prezentací (např. Microsoft PowerPoint, Google Slides, LibreOffice Impress).
   * Přidejte nové snímky s obsahem (text, obrázky, grafy).
2. **Přechody snímků:**
   * Vyberte snímek, na který chcete přidat přechod.
   * V panelu nástrojů vyberte Přechody (Transition).
   * Můžete zvolit různé efekty jako Fade, Push, Wipe apod.
   * Nastavení rychlosti přechodu a případně zvukového efektu.
3. **Závěrečná úprava prezentace:**
   * Kontrola gramatiky a pravopisu – využití funkce automatické kontroly.
   * Upravení designu – jednotný vzhled, barvy a písmo.
   * Přidání animací – pro texty a objekty na snímku.
4. **Způsoby uložení prezentace:**
   * Uložení v původním formátu (např. .pptx pro PowerPoint).
   * Uložení jako PDF pro snadné sdílení.

Uložení jako video (PowerPoint umožňuje exportovat prezentaci do video formátu pro automatické přehrávání).

**2. Vysvětlete co to je framework, jakým způsobem šetří programátorovi práci. Vysvětlete k čemu slouží framework click v programovacím jazyku Python a popište čím se programátor nemusí již zabývat, když tento framework využije.**

* **Co je to framework?**
* **Framework** je soubor knihoven a nástrojů, které usnadňují vývoj aplikací tím, že poskytují předem definovanou strukturu, nástroje a funkce.
* Umožňuje **rychlejší vývoj** aplikací, protože programátor se nemusí starat o opakující se činnosti (např. správa databází, šablony pro UI).
* Frameworky jsou obvykle navrženy pro určité typy aplikací (např. **webové aplikace**, **desktopové aplikace**).
* **Jakým způsobem framework šetří programátorovi práci?**
* **Zajišťuje opakovaně použitelné kódy** pro běžné operace (např. ověřování uživatelů, připojení k databázi).
* Poskytuje **standardizovanou strukturu** aplikace, což zjednodušuje údržbu a spolupráci.
* **Zkracuje čas vývoje** tím, že programátor nemusí vyvíjet základní funkce od nuly.
* **Framework Click v Pythonu**
* **Click** je framework pro **vývoj příkazových řádkových aplikací** v Pythonu.
* Umožňuje rychlou tvorbu **CLI (Command-Line Interface) aplikací**.
* **Co programátor nemusí řešit při použití Click:**
* **Zpracování argumentů a parametrů** – Click poskytuje jednoduché způsoby pro definici a zpracování argumentů příkazového řádku.
* **Nápověda pro uživatele** – automaticky generuje nápovědu pro příkazy a jejich parametry.
* **Správa chyb** – Click se postará o správné zobrazení chybových hlášení a validaci vstupů.
* **Příklad použití frameworku Click v Pythonu:**
* python
* ZkopírovatUpravit
* import click
* @click.command()
* @click.argument('name')
* def hello(name):
* click.echo(f"Hello, {name}!")
* if \_\_name\_\_ == '\_\_main\_\_':
* hello()

**3. Popište důvody použití virtuálního prostředí v jazyku Python. Ukažte jakými způsoby ho lze vytvořit (IDE Pycharm, terminal), jak se aktivuje, jakými způsoby do něj instalujeme moduly (IDE Pycharm, terminal). Popište k čemu slouží příkaz freeze instalátoru pip, soubor requirements.txt a jak nainstalujeme moduly do virtuálního prostředí z requirements.txt. Ukažte jak spustíme program v jazyku Python ve virtuálním prostředí.**

* **Izolace projektů: Každý projekt může mít své vlastní závislosti, což znamená, že instalované moduly v jednom projektu neovlivní jiný projekt.**
* **Zabránění konfliktům verzí: Každý projekt může používat jinou verzi knihovny, což zabraňuje konfliktům mezi verzemi.**
* **Snadná správa závislostí – umožňuje přesně určit, které verze knihoven projekt potřebuje.**

**Jak vytvořit virtuální prostředí?**

1. **Vytvoření virtuálního prostředí (v terminálu):**
   * **Použijte příkaz:**

**bash**

**ZkopírovatUpravit**

**python -m venv moje\_virt\_env**

* + **Tím se vytvoří složka moje\_virt\_env, která obsahuje izolované prostředí.**

1. **Aktivace virtuálního prostředí:**
   * **Na Windows:**

**bash**

**ZkopírovatUpravit**

**.\moje\_virt\_env\Scripts\activate**

* + **Na Linux/macOS:**

**bash**

**ZkopírovatUpravit**

**source moje\_virt\_env/bin/activate**

1. **Instalace modulů do virtuálního prostředí:**
   * **V PyCharm:**
     + **Přejděte do File → Settings → Project: [název projektu] → Python Interpreter.**
     + **Vyberte virtuální prostředí a klikněte na "+" pro instalaci balíčků.**
   * **V terminálu:**
     + **Použijte pip:**

**bash**

**ZkopírovatUpravit**

**pip install requests**

**Příkaz freeze a soubor requirements.txt:**

* **pip freeze: Zobrazí všechny nainstalované moduly a jejich verze v aktuálním virtuálním prostředí.**
  + **Příklad výstupu:**

**bash**

**ZkopírovatUpravit**

**requests==2.25.1**

**flask==1.1.2**

* **requirements.txt: Textový soubor, který obsahuje seznam všech závislostí projektu. Tento soubor lze použít pro instalaci stejných modulů na jiném počítači.**

**Instalace modulů z requirements.txt:**

* **Příkaz:**

**bash**

**ZkopírovatUpravit**

**pip install -r requirements.txt**

**Spuštění programu v Pythonu ve virtuálním prostředí:**

* **Jakmile máte aktivované virtuální prostředí, stačí spustit program jako obvykle:**

**bash**

**ZkopírovatUpravit**

**python muj\_program.py**