Ukol 1

# Application 12

**2. Problémy původního řešení**

**a) Špatné oddělení funkcí**

* Modul Fce\_Obdelnik měl dvě role:
  + **Výpočty** (pocitej\_obdelnik)
  + **Komunikace s uživatelem** (nacti\_strany\_obdelnika, tisk\_UdajeObdelnika)
* To porušovalo princip **Single Responsibility Principle (SRP)** – každý modul by měl mít jen jednu hlavní úlohu.

**b) Omezená znovupoužitelnost**

* Pokud bychom chtěli použít výpočty v aplikaci s grafickým rozhraním (GUI), museli bychom modifikovat Fce\_Obdelnik, což není správné.

**c) Zbytečné závislosti**

* Modul Tool\_Obdelnik musel zahrnovat jak Fce\_Obdelnik, tak Obdelnik\_IO, i když některé funkce mohly být lépe oddělené.

**3. Navržené řešení**

**a) Rozdělení modulů**

1. **Fce\_Obdelnik**
   * **Úloha**: Pouze matematické výpočty (obvod, obsah).
   * **Proč?** Aby mohl být použit v jakékoli aplikaci (konzole, GUI, web).
   * **Příklad**:

Une image contenant texte, logiciel, Logiciel multimédia, Police

Le contenu généré par l’IA peut être incorrect.

**2.Obdelnik\_IO**

* **Úloha**: Komunikace s uživatelem (načítání vstupů, tisk výsledků).
* **Proč?** Pokud změníme způsob vstupu/výstupu (např. na GUI), stačí upravit jen tento modul.
* **Příklad**:

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

3. **Tool\_Obdelnik**

* **Úloha**: Propojuje logiku (Fce\_Obdelnik) a uživatelské rozhraní (Obdelnik\_IO).
* **Proč?** Aby hlavní program (Application\_12.cpp) nemusel znát detaily výpočtů a I/O operací.

Une image contenant texte, capture d’écran, logiciel, affichage

Le contenu généré par l’IA peut être incorrect.

Shema zavislosti

Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.

**4. Co mohlo způsobit problémy?**

**a) Nesprávné propojení modulů**

* Pokud by Fce\_Obdelnik stále obsahoval I/O funkce, změna rozhraní by vyžadovala úpravu i tohoto modulu → **porušení modularity**.

**b) Cyklické závislosti**

* Pokud by Fce\_Obdelnik volal Obdelnik\_IO a zároveň Obdelnik\_IO volal Fce\_Obdelnik, vznikl by **nekonečný cyklus**.

**c) Chybějící oddělení překladu**

* Pokud by všechny funkce byly v jednom .cpp souboru, každá změna by znamenala **překompilování celého projektu** → ztráta výhody modulárního překladu.

**5. Pozorování překladu (bod b zadání)**

* Při změně **jednoho modulu** (např. Fce\_Obdelnik.cpp) se překompiluje **jen jeho .obj soubor**.
* Ostatní .obj soubory zůstávají nezměněné (např. Obdelnik\_IO.obj).
* **Závěr**: Modulární překlad šetří čas, protože kompilátor překládá **pouze změněné části**.

Vystup

Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

# Application 14

Oproti předchozí verzi (Application\_12) je zde použita struktura Obdelnik, která sdružuje všechny potřebné hodnoty (strany, obvod, obsah)

**2. Hlavní změny oproti Application\_12**

* **Použití struktury Obdelnik** místo samostatných proměnných.
* **Čistší oddělení modulů** – výpočty, I/O operace a hlavní logika jsou lépe oddělené.
* **Vylepšená vizualizace závislostí** pomocí funkce zobrazZavislosti().

**Proč je toto řešení lepší?**

1. **Struktura Obdelnik zjednodušuje předávání dat**
   * Místo 4 samostatných proměnných (a, b, o, S) se předává jen jedna struktura.
   * Výhoda: Kód je přehlednější a méně náchylný k chybám (např. špatné pořadí argumentů).
2. **Lepší oddělení zodpovědností**
   * Fce\_Obdelnik = **výpočty** (jen matematika, žádný vstup/výstup).
   * Obdelnik\_IO = **komunikace s uživatelem** (načítání a tisk).
   * Tool\_Obdelnik = **propojení logiky a I/O**.
3. **Snadná rozšiřitelnost**
   * Pokud budeme chtít přidat další vlastnosti obdélníka (např. úhlopříčku), stačí upravit **pouze strukturu Obdelnik a Fce\_Obdelnik**, aniž bychom museli měnit I/O modul.

**4. Možné problémy a jejich řešení**

**a) Problém: Cyklické závislosti**

* **Původní problém**:  
  Obdelnik\_IO.h includuje Fce\_Obdelnik.h (kvůli struktuře Obdelnik), ale Fce\_Obdelnik by neměl záviset na I/O.
* **Řešení**:  
  Vytvořit **samostatný hlavičkový soubor Obdelnik.h** pouze pro strukturu, aby se zabránilo kruhové závislosti.  
  *(V tomto projektu to není kritické, ale u větších projektů by to mohlo způsobit problémy.)*

**b) Problém: Překladové závislosti**

* Pokud bychom změnili Fce\_Obdelnik.cpp, překompiluje se jen jeho .obj soubor.
* Pokud ale změníme Obdelnik.h, překompilují se **všechny soubory, které ji používají** (Fce\_Obdelnik.cpp, Obdelnik\_IO.cpp, Tool\_Obdelnik.cpp).
* **Doporučení**: Hlavičkové soubory by měly obsahovat **minimum kódu**, aby se omezilo zbytečné překompilování.

**c) Problém: Bezpečnost vstupů**

* Funkce nacti\_strany\_obdelnika() používá scanf(), která **nekontroluje platnost vstupu** (např. záporné hodnoty).
* **Vylepšení**: Přidat validaci:

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

Vystup

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.

# MathApp & MyMathLibrary

**2. Struktura projektu**

Projekt se skládá ze dvou hlavních částí:

1. **Vlastní knihovna MyMathLibrary** - obsahuje matematické funkce
2. **Aplikační projekt MathApp** - demonstruje použití knihovny

**3. Podrobný popis kódu**

**3.1 Vlastní knihovna (MyMathLibrary)**

**Une image contenant texte, capture d’écran, logiciel, affichage

Le contenu généré par l’IA peut être incorrect.**

**Klíčové vlastnosti:**

* **Statická metoda** Factorial - lze volat bez instance třídy
* **Kontrola vstupu** - vyvolá výjimku pro záporná čísla
* **Iterativní výpočet** - efektivnější než rekurzivní přístup
* **XML komentáře** - umožňují IntelliSense v IDE

**3.2 Hlavní aplikace (MathApp)**

Une image contenant texte, capture d’écran, logiciel

Le contenu généré par l’IA peut être incorrect.

Une image contenant texte, capture d’écran, logiciel, affichage

Le contenu généré par l’IA peut être incorrect.

**Klíčové části aplikace:**

1. **Načítání vstupu od uživatele**
   * Použití TryParse pro bezpečné zpracování vstupu
   * Ošetření neplatných vstupů
2. **Použití vlastní knihovny**
   * Volání MathFunctions.Factorial()
   * Zachytávání výjimek pro záporná čísla
3. **Použití standardní matematické knihovny**
   * Převod stupňů na radiány
   * Výpočet sinus pomocí Math.Sin()

**4. Možná rozšíření**

1. **Přidání dalších matematických funkcí:**
   * Kombinační čísla
   * Fibonacciho posloupnost
   * Prvočíselné testy
2. **Vylepšení výpočtu faktoriálu:**
   * Rekurzivní implementace
   * Optimalizace pro velká čísla

**5. Návod k použití**

1. **Zkompilování knihovny:**
   * Vytvořte projekt typu "Class Library"
   * Přejmenujte namespace na MyMathLibrary
   * Přidejte metodu Factorial
2. **Použití knihovny v aplikaci:**
   * Přidejte referenci na knihovnu
   * Importujte namespace pomocí using MyMathLibrary
   * Volání metod knihovny

Vystup

Une image contenant texte, capture d’écran, Police

Le contenu généré par l’IA peut être incorrect.