

## Freiwillige Offline-Aufgabe 09-01 (INF & WI):

**Templates: Funktion swap\_vars()**

*(geübte C++ Konstrukte: Funktions-Templates & deren explizite Instanziierung)*

Schreiben Sie eine Template-Funktion `void swap_vars(...)`, welche die Werte ihrer zwei Parametervariablen tauscht.

Die Funktion `swap_vars()` soll daher zwei Parameter `x1` und `x2` haben vom identischen, allgemein gehaltenen Typ `T`.

Schreiben Sie für den Prototypen der Template-Funktion eine geeignete, vollständige Headerdatei `swap_vars.h`. Schreiben Sie ferner eine Implementierungsdatei `swap_vars.cpp`, welche die vollständige Realisierung der Funktion sowie die *explizite Instanziierung* für `int` und für `string` realisiert.

Programmieren Sie darüber hinaus in einer Datei `main.cpp` ein Hauptprogramm, welches die angegebenen Testläufe realisiert.

**Kommentar:** In diese Datei brauchen (wie auch in der Vorlesung dargestellt) *keine extern* Anweisungen zur Verhinderung einer zusätzlichen impliziten Instanziierung eingefügt zu werden: Da der Code in `main.cpp` nur die Headerdatei inkludiert und dort nur der **Prototyp** der Template-Funktion steht, "bleibt der Rumpf dieser Funktion unbekannt" und der Code in `main.cpp` kann gar keine implizite Instanziierung durchführen. *extern* Anweisungen wären also unnötig und führen bei der Art, wie wir aktuell die Jenkins-Tests für diese Aufgabe programmiert haben, sogar zu Fehlermeldungen ... Ein Vorteil der **expliziten** Instanziierung (auf die beschriebene Art) ist ja insbesondere, dass man sich diese *extern* Anweisungen sparen kann ...

**Achtung!** Im Hauptprogramm werden Sie zuerst zwei `int` Eingaben mittels `cin >> ...` entgegennehmen, dann zwei `string` Eingaben mittels `getline()`. Programmieren Sie daher auf jeden Fall hinter die zweite `cin >> ...` Eingabe (und wenn Sie es systematisch machen wollen hinter jede `cin >> ...` Eingabe) den Befehl `cin.ignore()`, sonst funktioniert die anschließende (erste) `getline()` Eingabe nicht wie gedacht, da noch ein Newline (Zeilenumbruch) auf der Eingabe verbleibt.  
Siehe GIP-INF Übung am 8.11.2019!

## Testläufe (Benutzereingaben zur Verdeutlichung unterstrichen):

---

```
Bitte geben Sie die erste int Zahl ein: ? 1
Bitte geben Sie die zweite int Zahl ein: ? 22
Erste Zahl nachher: 22
Zweite Zahl nachher: 1
Bitte geben Sie den ersten String ein: ? a
Bitte geben Sie den zweiten String ein: ? bb
Erster String nachher: bb
Zweiter String nachher: a
Drücken Sie eine beliebige Taste . . .
```

---

```
Bitte geben Sie die erste int Zahl ein: ? 333
Bitte geben Sie die zweite int Zahl ein: ? 4
Erste Zahl nachher: 4
Zweite Zahl nachher: 333
Bitte geben Sie den ersten String ein: ? ab cd
Bitte geben Sie den zweiten String ein: ? XY Z
Erster String nachher: XY Z
Zweiter String nachher: ab cd
Drücken Sie eine beliebige Taste . . .
```

---

```
Bitte geben Sie die erste int Zahl ein: ? 1
Bitte geben Sie die zweite int Zahl ein: ? 2
Erste Zahl nachher: 2
Zweite Zahl nachher: 1
Bitte geben Sie den ersten String ein: ? ab cd
Bitte geben Sie den zweiten String ein: ? (leerer Eingabestring)
Erster String nachher: (leere Ausgabe, aber ein Leerzeichen hinter dem Doppelpunkt!)
Zweiter String nachher: ab cd
Drücken Sie eine beliebige Taste . . .
```

---