**WPF „Python“**
**Übungsaufgaben 28.01.2025**

1.) Implementieren Sie eine Klasse `Stack` und bauen Sie dabei auf der folgenden Definition auf:

```
class Stack:
def __init__(self):
 self.st = [ ]
  …
```

Implementieren Sie die folgenden Methoden:

a) Implementieren Sie eine Methode `push`, die Elemente auf den Stack lädt.
b) Implementieren Sie eine Methode `pop`, die das oberste Element vom Stack löscht und dieses Element zurückliefert.
c) Implementieren Sie die Methode `__len__`, die die Anzahl der Elemente des Stacks zurückliefert.
d) Implementieren Sie eine Methode `toList`, die alle im Stack enthaltenen Elemente als Liste zurückliefert.
e) Implementieren Sie eine Methode `multPop`, die eine Ganzzahl übergeben bekommt und n Pop-Operationen ausführt. Die Ergebnisse aller Pop-Operationen sollen als Liste zurückgeliefert werden.

Beispiel für die Funktionsweise des Stacks:

```
>>> s = Stack()
>>> s.push(1); s.push(“Hallo”); s.push(4.32); s.push(True)
>>> s.pop()
True
>>> s.push(“a”)
>>> s.pop(); s.pop()
(‘a’, 4.32)
>>> s.pop()
‘Hallo’
```

2.) Implementieren Sie eine Klasse für einen Aufgaben-Manager, der anstehende Aufgaben (jeweils repräsentiert als String, der die Aufgabenbeschreibung enthält) verwaltet. Jede Aufgabe hat eine Priorität (repräsentiert als eine Ganzzahl), die die Wichtigkeit oder Dringlichkeit der Aufgabe angeben soll. Je kleiner diese Zahl, desto wichtiger die entsprechende Aufgabe. Für eine Priorität kann es mehrere Aufgaben geben. Als Container-Klasse für die Aufgaben einer bestimmten Priorität soll die unter 1.) implementierte Klasse dienen. Der „Kopf“ der Klasse sei gegeben durch:

```
class AufgabenManager:
def __init__(self):
 self.aufgaben = {}
…
```

Implementieren Sie die folgenden Methoden, so dass die Klasse gemäß der am Ende der Aufgabenstellung gezeigten Beispielanwendung funktioniert.

a) Die Methode `neueAufgabe`, die eine neue Aufgabe mit einer bestimmten Priorität hinzufügt. Der Methode muss ein String und eine Ganzzahl übergeben werden.
b) Die Methode `hoechstePrio`, die die höchste Prioritätsstufe zurückliefern soll.
c) Die Methode `erledigeNaechsteAufgabe`, die die nächste Aufgab der höchsten Prioritätsstufe erledigt, d.h. aus der Datenstruktur löscht und den Aufgaben-String zurückliefert.
d) Eine Methode `alleAufgabenMitPrio`, die alle Aufgaben einer bestimmten Prioritätsstufe zurückliefert. Die Methode soll eine (evtl. auch leere) Liste von Strings zurückliefern.
e) Die Methode `allePrios`, die eine Liste alle Prioritätsstufen zurückliefert.
f) Die Methode `anzahlAufgabenPrio`, die die Anzahl der Aufgaben einer bestimmten Prioritätsstufe zurückliefern soll. Die Methode soll eine Prioritätsstufe übergeben bekommen.
g) Die Methode `anzahlAufgaben`, die zurückliefern soll, wie viele Aufgaben es insgesamt gibt.

Hier eine Beispielanwendung der Klasse:

```
>>> aufgs = AufgabenManager()
>>> aufgs.neueAufgabe(„Kueche putzen“, 5)
>>> aufgs.neueAufgabe(„Auf Prog 1 lernen“, 1)
>>> aufgs.neueAufgabe(„Oma besuchen“, 2)
>>> aufgs.neueAufgabe(„Auf Mathe 1 lernen“, 1)
>>> aufgs.neueAufgabe(„Fahrrad putzen“, 10)
>>> aufgs.erledigeNaechsteAufgabe ()
„Auf Mathe 1 lernen“
>>> aufgs.erledigeNaechsteAufgabe ()
„Auf Prog 1 lernen“
>>> print(aufgs.hoechstePrio())
2
>>> print(aufgs.anzahlAufgabenPrio())
0
>>> print(anzahlAufgaben())
3
```