Hochschule
Albstadt-Sigmaringen
Albstadt-Sigmaringen University

Studiengänge ITS,TI,WIN
Vorlesung: **Prof.Dr.Derk Rembold**
Praktikum: Ralf Mispelhorn, Dominik Höpfl

Programmieren 1

Aufgabenblatt 6

Abgabe: Die Lösungen sollten zeitnah als IPYNB-Datei fertiggestellt werden. Der Code muss ausreichend kommentiert sein und die Variablen müssen sinnvoll benannt werden. Sie müssen die Aufgabe selbst programmiert haben. Sie können Fragen in Form von Kommentaren im Code stellen, falls etwas nicht funktioniert hat. Die Antwort erfolgt dann im Praktikum mündlich. Sie dürfen nie mehr als drei Aufgabenblätter im Rückstand sein.

Hilfsmittel: Kein Copy-Paste aus dem Internet, alles muss selbstständig programmiert sein. Sie dürfen die IPython-Notebook-Skripte aus der Vorlesung (liegen nach der jeweiligen Vorlesung auf Ilias) und die Python-Einführung auf Ilias verwenden. Außerdem ist die Hilfe-Funktion `help(...)` und die Methode `dir(...)` zur Auflistung der verfügbaren Funktionen zu empfehlen.

Anwesenheit: Grundsätzlich herrscht Anwesenheitspflicht. Ein Attest ist notwendig, wenn jemand nicht kommen kann. Bei unentschuldigtem Fehlen ist das Praktikum nicht bestanden. Von der Teilnahme an der Klausur wird in diesem Fall dringend abgeraten.

Aufgabe 6.0

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 `n` 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'
```

Aufgabe 6.1

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 / Dringlichkeit der Aufgabe angeben soll. Je kleiner diese Zahl, desto wichtiger die entsprechende Aufgabe. Für eine Priorität kann es mehrere Aufgaben geben.

Zum Speichern der Aufgaben soll pro Priorität die oben implementierte Klasse `Stack` 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 Aufgabe 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 (eventuell auch leere) Liste von Strings zurückliefern.

(e) Die Methode `allePrios`, die eine Liste aller Prioritätsstufen zurückliefern soll.

(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(1))
0
>>> print(aufgs.anzahlAufgaben())
3
```