SoSe 2019

Prof. Dr. Margarita Esponda

Objektorientierte Programmierung

2. Übungsblatt

Ziel: Auseinandersetzung mit Rekursion vs. Iteration und die Anwendung von Wörterbüchern (Dictionaries) in Python.

1. Aufgabe (4 Punkte)

Definieren Sie eine Funktion, die bei Eingabe einer natürlichen Zahl **n** die Liste aller (a, b, c) Pythagoras-Zahlentripel mit $a^2 + b^2 = c^2$, $0 < a \leq b \leq c \leq n$ und ohne Wiederholungen, berechnet.

Anwendungsbeispiel:

```
>>> pythTripelsSmaller 16
[(3,4,5), (5,12,13), (6,8,10), (9,12,15)]
```

2. Aufgabe (4 Punkte)

Definieren Sie eine Python-Funktion **apply_until**, die als Eingabe eine einstellige Funktion **f**, ein Prädikat **p** (Funktion) und eine Liste **xs** bekommt und die Funktion **f** auf die Elemente der Liste anwendet, solange das Prädikat **p** erfüllt wird. Das bedeutet, dass die Anwendung der Funktion **f** auf die Elemente der Liste beim ersten Vorkommen eines Elements, das **p** nicht erfüllt, gestoppt wird.

Anwendungsbeispiel:

```
>>> apply_until ( factorial, odd,  [3, 5, 7, 4, 9, 6] )
[6, 120, 5040]
```

Zum Testen verwenden Sie die **factorial** und **odd** Funktionen aus den Vorlesungsfolien.

3. Aufgabe (4 Punkte)

Betrachten Sie folgende Haskell-Funktion höherer Ordnung:

```
filter p [] = []
filter p (x:xs) | p x         = x: filter p xs
                | otherwise = filter p xs
```

Die **filter** Funktion bekommt ein Prädikat **p** (eine Funktion, die als Rückgabewert nur Wahrheitswerte zurückgibt) und eine Liste als Argumente und berechnet die Teilliste aller Elemente, die das Prädikat **p** erfüllen.

a) Programmieren Sie eine rekursive Python-Funktion **filter_rec**, die genau der gleichen Semantik entspricht.

Anwendungsbeispiel:

```
>>> filter_rec ( odd, [2, 4, 3, 7, 1, 0, 8, 3] )
[3, 7, 1, 3]
```

b) Programmieren Sie eine nicht rekursive Version der Funktion (**filter_iter**).

4. Aufgabe (4 Punkte)

Schreiben Sie eine **repeat** Python-Funktion, die die Funktion **randint** des **random**-Moduls von Python benutzt, und berechnet, nach wie vielen Zufallszahlen die erste Wiederholung vorkommt. Die Zufallszahlen der **randint** Funktion sollten in einem Wertebereich zwischen **1** und **1000000** liegen.

a) Verwenden Sie dafür eine Dictionary-Datenstruktur.

b) Sie müssen die zwei Argumente für die **randint**-Funktion, die dem Wertebereich entsprechen, in Ihre **repeat** Funktion miteingeben.

c) Der Rückgabewert soll ein Tupel mit der Liste der generierten Zahlen und die Anzahl der Zahlen sein, bevor die erste Wiederholung vorkommt.

Anwendungsbeispiel:

>>> repeat (1, 1000000)

([299442, 426471, 467326, 378698, 866095, ...], 938)

5. Aufgabe (12 Punkte) Geburtstagsparadox

Die Wahrscheinlichkeit, dass 2 Personen am gleichen Tag Geburtstag haben, kann mit Hilfe eines Zufallsgenerators empirisch approximiert werden.

a) Schreiben Sie eine Funktion **double_birthday**, die zufällige Geburtstage produziert und zählt, nach wie vielen Versuchen eine Wiederholung vorkommt. Verwenden Sie für die Zwischenspeicherung der Daten ein Wörterbuch (Dictionary) als Datenstruktur.

b) Programmieren Sie eine zweite Funktion **repeat_double_birthday**, die das Experiment lange genug wiederholt und mit Hilfe eines Arrays aufzählt, nach wie vielen Versuchen (2, 3, 4,...,bis max. 367) eine Wiederholung vorkommt.

c) Mit der Ergebnisliste aus b) können Sie dann eine Funktion **birthday_paradox** programmieren, die die Wahrscheinlichkeit, dass auf einer Party mit **n** Gästen zwei Personen am gleichen Tag Geburtstag haben, empirisch berechnet.

5. Aufgabe (2 Punkte)

Schreiben Sie getrennte Test-Funktionen für alle 5 Aufgaben für Ihren Tutor.

Wichtige Hinweise:

1) Verwenden Sie geeignete Namen für Ihre Variablen und Funktionen, die den semantischen Inhalt der Variablen oder die Funktionalität der Funktionen darstellen.
2) Verwenden Sie vorgegebene Funktionsnamen, falls diese angegeben werden.
3) Kommentieren Sie Ihre Programme.
4) Verwenden Sie geeignete Hilfsvariablen und Hilfsfunktionen in Ihren Programmen.
5) Löschen Sie alle Programmzeilen und Variablen, die nicht verwendet werden.
6) Schreiben Sie getrennte Test-Funktionen für alle 4 Aufgaben für Ihren Tutor.
7) Die Lösungen sollen in Papierform und elektronisch (KVV-Upload) abgegeben werden.