Programmierung mit Python	8 - D 2 - KA-L 3 (i)	Gr - 2010/2011 - 30.03.11

Aufgabe 1 (Grundbegriffe im Rahmen der Programmierung)

Erläutere die folgenden Begriffe:

(a) **Programmierung**

Unter der Programmierung versteht man die Umsetzung von Entwürfen, Planungen etc. für ein gewünschtes Computerprogramm am Computer. Entsprechende Anweisungen für den Computer werden in einer Programmiersprache formuliert. Im weitesten Sinne zählen auch bereits die Vorüberlegungen zur Programmierung (Planungsprozess).

(b) **Programmiersprache**

Unter einer Programmiersprache versteht man eine künstliche Sprache zur Verständigung zwischen Mensch und Maschine. Jede Programmiersprache hat ihre eigenen Regeln, genannt Syntax, zur Formulierung von Programmen in dieser Programmiersprache. Die in einer Programmiersprache formulierten Anweisungen sind für Menschen einfach verständlich. Der Interpreter kann diese einfach in Maschinensprache übersetzen und zur Ausführung an den Prozessor übergeben.

(c) **Kommentare (Kopfkommentar, normaler Kommentar und Docstring)**

Kommentare dienen dazu, das Programm oder einzelne Anweisungen bzw. den Effekt oder Sinn der Anweisungen für den menschlichen Leser zu verdeutlichen. Es soll das Ziel verfolgt werden, durch diese zusätzlichen „Erklärungen“ das Programm verständlicher zu machen.
Normale oder reguläre Kommentare könne sich auf einzelne Zeilen beziehen. Diese beginnen dann mit dem #-Symbol.
Mit dem Kopfkommentar verfolgt man das Ziel, Zusatzinformationen über das Programm zu geben, insbesondere eine kurze, aber verständliche Beschreibung des Zweckes des Programmes.
Sogenannte Docstrings dienen zur Beschreibung von Funktionen. Diese werden bereits während der Programmierung angezeigt. Auf diese Weise kann man sicherstellen, dass man die richtige Funktion nutzt.

Aufgabe 2 (Ausgaben und Datentypen)

Erläutere die Ausgaben der folgenden Befehle.
Gehe dabei insbesondere auf die Unterschiede in den Argumenten der `print()`-Funktion ein.

(a)

```
1 # print("Viel Erfolg!)
```

Ausgabe:
In diesem Fall erfolgt keine Ausgabe, da es sich um einen Kommentar handelt. Ferner wäre der Ausdruck syntaktisch nicht korrekt, da die schließenden Anführungszeichen bei der Zeichenkette fehlen.

(b)

```
1 print(2)
```

Ausgabe: 2
Es wird die Zahl 2 ausgegeben. Das Argument der Funktion ist eine Zahl.

(c)

```
1 for zwei in range(2):
2         print("zwei")
```

Ausgabe: zwei [Zeilenumbruch] zwei
Die Schleife wird zweimal `range(2)` durchlaufen. In jedem Durchgang wird die Zeichenkette `"zwei"` ausgegeben. Zu beachten ist hier, dass in der `print()`-Funktion nicht die Schleifenvariable verwendet wird, sondern es sich um eine Zeichenkette handelt, die dem Variablennamen entspricht.

(d)

```
1 print("2"+"18")
```

Ausgabe: 218
Es handelt sich hier um zwei Zeichenketten, die vor der Ausgabe aneinander gehängt werden. Es entsteht also die Zeichenkette `"218"`, die als Zahl interpretiert werden kann. Zu beachten ist, dass hier keine Addition stattfindet.

(e)

```
1 def myFunction():
2         print("Hallo Welt")
```

Ausgabe:
Hier erfolgt wieder keine Ausgabe. Es wird eine Funktion definiert, die eine Ausgabe hat. Die Funktion wird aber nicht aufgerufen, so dass keine Ausgabe erfolgt. Erst wenn man die Funktion durch die Anweisung `myFunction()` aufruft, erfolgt die Ausgabe der Zeichenkette `"Hallo Welt"`.

Aufgabe 3 (Elemente der Programmierung)

(a) Stelle kurz dar, zu welchem Zweck im Rahmen der Programmierung die Kontrollstruktur Schleife eingesetzt wird und zwischen welchen beiden grundlegenden Arten man dabei unterscheidet.

Die Programmierer setzen die Kontrollstruktur Schleife ein, um bestimmte Folgen von Anweisungen auf einfache Weise wiederholt ausführen zu können. Die entsprechende Folge von Anweisungen bezeichnet man auch als Schleifenkörper. Die „Definition“ der Schleife erfolgt im Schleifenkopf. Man unterscheidet zwischen der

- Zählschleife.
  Hier werden die festgelegten Anweisungen so oft wiederholt, wie es im Vorfeld festgelegt wurde.
- bedingten Schleife.
  Hier werden die Anweisungen wiederholt, solange eine bestimmte Bedingung erfüllt ist, beispielsweise „Der Nutzer drückt die Taste *a*“.

(b) In einem Programm benötigst du an unterschiedlichen Stellen eine bestimmte Folge von Befehlen. Es ist möglich, diese Befehle an allen entsprechenden Stellen in das Programm zu schreiben. Wir haben noch eine zweite Möglichkeit kennengelernt, dies zu realisieren.
Erläutere diese kurz und stelle mindestens einen Vorteil dieser Realisierung dar.

Man kann eine Funktion definieren (Funktionskopf), die bei Ausführung die entsprechenden Anweisungen (Funktionskörper) ausführt.
Vorteile:

- einfach lesbarer Quelltext
- größere bzw. höhere Übersichtlichkeit
- kürzerer Quelltext
- Fehler müssen nur noch an einer Stelle korrigiert werden
- einfache Wiederverwendung von Programmteilen
- dadurch wird die Reaktion auf Ereignisse möglich

(c) Erläutere die folgende Anweisung: `anzahl=100`

Der Variablen `anzahl` wird der Zahlwert `100` zugewiesen. Wir können jetzt die Variable bzw. den zugehörigen Wert in den späteren Anweisungen nutzen. Beispielsweise würde nun `print(anzahl)` den Wert `100` ausgeben.

(d) Gib Beispiele für die Anweisungen an, die vor der folgenden Anweisung ausgeführt werden müssen: `a=b+c+2`
Begründe deine Entscheidung.

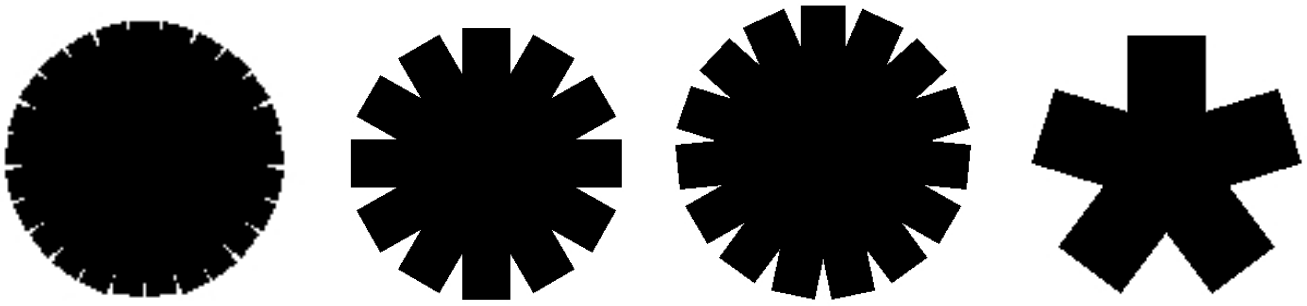
Die in der Zuweisung verwendeten Variablen müssen im Vorfeld einen Wert zugewiesen bekommen haben, um sie in dieser Anweisung nutzen zu können.
Beispielsweise sind `c=17` und `b=100` möglich.

(e) Schreibe eine Funktion `quadratzahlen()`, die mittels einer Schleife das Quadrat der Zahlen 0 bis 99, also das Quadrat aller maximal zweistelligen Zahlen, ausgibt.

```
1 def quadratzahlen():
2         """ gibt alle maximal zweistelligen Quadratzahlen aus """
3         for quadratzahl in range(100):
4                 print(quadratzahl**2)
5         # Schleifenende
6 # Funktionsende
```

Aufgabe 4 (Zeichnen von „Zahnrädern“)

In dieser Aufgabe geht es um die schrittweise Konstruktion eines Programmes, dass in der Lage ist, diese und ähnliche „Zahnräder“ zu zeichnen. Die „Zahnräder“ sollen hinsichtlich der Anzahl an Zähnen und Abmessungen der einzelnen „quadratischen“ Zähnen so gestaltet sein, dass diese leicht zu ändern sind.



(a) Erstelle einen geeigneten Kopfkommentar für das Programm.

```
1 # Kopfkommentar
2 # Autor: Christian Graf
3 # Datum: 25.03.2011
4 # Beschreibung: Programm zum Zeichnen von variablen Zahnrädern
```

(b) Um entsprechende Zeichnungen erstellen zu können, müssen wir u.a. ein Objekt der Klasse Turtle erzeugen. Treffe auch die dazu nötigen Vorbereitungen. Lege ebenfalls Variablen für die Zahnanzahl und die zugehörige Abmessung an. Die zugehörigen Variablen und Werte darfst du frei wählen.

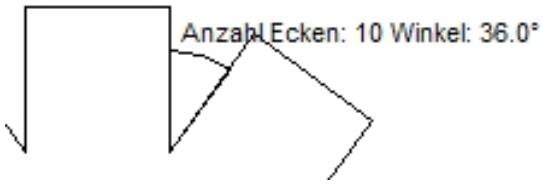
```
1 from turtle import Turtle
2 myTurtle = Turtle("turtle")
```

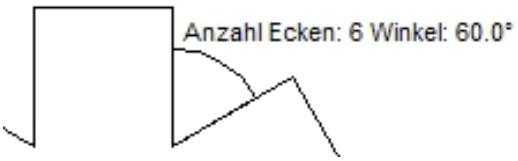
```
3 seitenlaenge=100
4 anzahl=10
```

(c) Für das Zeichnen eines einzelnen Zahnes soll eine entsprechende Funktion geschrieben werden. Die Turtle soll am Ende wieder in die ursprüngliche Richtung gedreht sein. Schreibe die entsprechende Funktion inkl. Funktionskopf etc.

```
1  def zahn():
2      """ zeichnet eine einzelne Ecke des Zahnrades """
3      # Lösung mittels einer Schleife ist ebenfalls möglich
4      myTurtle.left(90)
5      myTurtle.forward(seitenlaenge)
6      myTurtle.right(90)
7      myTurtle.forward(seitenlaenge)
8      myTurtle.right(90)
9      myTurtle.forward(seitenlaenge)
10     myTurtle.left(90)
11 # Funktionsende
```

(d) Abschließend muss nun der Programmausschnitt geschrieben werden, der mittels der programmierten Funktion und einer weiteren geeigneten Kontrollstruktur das gesamte Zahnrad zeichnet. Zur Bestimmung des Drehwinkels zwischen zwei Zähnen helfen dir die beiden Abbildungen. Achte darauf, dass sich das Turtle-Objekt in die richtige Richtung dreht.





Hinweis: Fertige zur Unterstützung eine Skizze an, in der du bekannte Winkel markierst. Überlege genau, in welche Richtung die Turtle nach dem Zeichnen des ersten Zahnes gedreht ist und in welche Richtung eine Drehung erfolgen muss, damit der nächste Zahn gezeichnet werden kann. Ergänze zur weiteren Unterstützung Hilfslinien. Auf der nächsten Seite findest du weiteren Platz zur Bearbeitung.

```
1 for i in range(anzahl):
2         zahn()
3         myTurtle.right(360/anzahl)
4 # Schleifenende
```

(e) Erweitere den letzten Aufgabenteil so, dass das Zahnrad ausgefüllt wird. Nutze dazu die Methoden `begin_fill()` und `end_fill()`, die jedes Turtle-Objekt bereitstellt. Ferner soll das Objekt der Klasse Turtle am Programmende unsichtbar werden. Rufe dazu die Methode `hideturtle()` des Objektes auf.

```
1 myTurtle.begin_fill()
2 for i in range(anzahl):
3         zahn()
4         myTurtle.right(360/anzahl)
5 # Schleifenende
6 myTurtle.end_fill()
7 myTurtle.hideturtle()
```

Programmierung mit Python	8 - D 2 - KA-L 3 (v)	Gr - 2010/2011 - 30.03.11

Aufgabe 5 (Single-Choice Fragen)

Entscheide, ob die folgenden Aussagen richtig oder falsch sind.

	r	f	**Aussage** [Objekte etc.]
	☐	X	Ein Objekt ist der Bauplan einer Klasse.
	☐	X	Eine grundlegende Art der Programmierung ist die sogenannte variablen-orientierte Programmierung.
	X	☐	Bei der Objektorientierung versucht man die Realität möglichst detailgetreu in ein Programm bzw. dessen Entwurf umzusetzen.
	☐	X	Attribute von Objekten kann man nicht verändern.
	X	☐	Mit Methoden bezeichnen die Informatiker Fähigkeiten von Objekten.
	X	☐	Bevor man ein Objekt verwenden kann, muss man es erzeugen.
	☐	X	Sicherheitshalber sollte man alle Klassen laden, denn man kann ja nie wissen, was man später im Programm noch alles benötigt.
	X	☐	Jedes Objekt der Klasse Mensch hat ein Attribut „Gewicht“.

	r	f	**Aussage** [Schleifen und Funktionen etc.]
	X	☐	Mittels Kontrollstrukturen wird es möglich, den Programmablauf zu steuern.
	☐	X	Das reservierte Wort `def` benötigt man in einem Schleifenkopf.
	☐	X	Ein Funktionskopf muss immer eingerückt werden.
	X	☐	Mittels eines sog. Docstrings kommentiert man selbst geschriebene Funktionen.
	X	☐	Anweisungen in Schleifen- und Funktionskörpern werden eingerückt.

	r	f	**Aussage** [Variablen etc.]
	X	☐	Durch den Einsatz von Variablen kann man bestimmte Werte an unterschiedlichen Stellen im Programm verwenden.
	☐	X	`abcd` ist ein sinnvoller Variablenname für ein Turtle-Objekt.
	X	☐	`"Hallo"` ist eine Zeichenkette.
	☐	X	In Variablen sind Leerzeichen erlaubt.
	☐	X	Variablennamen müssen mit einer Zahl beginnen.
	X	☐	Das Zeichnen = nennt man in Python den Zuweisungsoperator.

	r	f	**Aussage** [Turtle etc.]
	☐	X	Nach Ausführung der Anweisung `heinz.left(360)` schaut das Turtle-Objekt `heinz` in eine andere Richtung.
	X	☐	Die Anweisungen `heinz.left(45)` und `heinz.right(315)` haben die gleichen Auswirkungen auf das Turtle-Objekt `heinz`.
	X	☐	Das Anfertigen einer Skizze hilft bei der Erstellung von Zeichnungen mit einem Objekt der Klasse Turtle.

	r	f	**Aussage** [Python etc.]
	☐	X	Das Tupel `range(100)` enthält die Zahlen 1 bis 100.
	☐	X	In Python können Anweisungen beliebig eingerückt werden.
	☐	X	Das Tupel `range(20)` enthält 19 Werte.
	X	☐	Kommentare beginnen in Python mit dem Zeichen #.
	X	☐	Die einzelnen Zeilen in einem Programm müssen syntaktisch korrekt sein.
	X	☐	In der PythonShell kann man Programmideen oder Teile von Programmen einfach testen.