Marvin Glaser
4424114

Gruppe 9
Wilhelm Esser

Programmieren 1 (PRG1) Übung 1

1.1

(a) Die van Neumann Architektur beschreibt einen Computer als ein Gerät aus fünf Funktionseinheiten. Diese Einheiten sind: Speicher (Ablage für Daten und Programme), Steuerwerk (koordiniert Aktionen der einzelnen Funktionseinheiten und interpretiert Programme), Rechenwerk (führt Operationen der interpretierten Programme aus), Eingabewerk und Ausgabewerk (Ein-, bzw. Ausgabepunkt von Daten). Ein wichtiges Charakteristikum ist, dass sowohl Daten, als auch Programme im Speicher abgelegt sind (keine getrennten Befehls- und Datenspeicher).

Die Neumann Architektur garantiert einen sequentiellen Ablauf von Aktionen, was die Programmierung erleichtert. Allerdings sorgt die Ablage von Programmen und Daten im selben Speicher dafür, dass zu einer gegebenen Zeit nur Programme oder Daten abgerufen werden können, nicht beides gleichzeitig. Dies führt dazu, dass die Datenübertragungsrate zwischen Speicher, Steuer- und Rechenwerk zum Geschwindigkeitsbestimmenden Schritt einer Datenverarbeitung werden (*Neumann bottleneck*). Die Harvard Architektur versucht dieses Problem zu umgehen, indem sie Programme und Daten in verschiedenen Speichern ablegt. Dies ermöglicht zwar den gleichzeitigen Zugriff auf Programme und Daten, allerdings geht hierbei auch der sequentielle Ablauf verloren, was dazu führen kann, dass die Ergebnisse von Operationen davon abhängen, welche Daten (Programme oder Daten) zuerst geladen werden (r*ace conditions*).

(Quelle: https://de.wikipedia.org/wiki/Von-Neumann-Architektur#Eigenschaften; Datum: 22.10.17; Uhrzeit: 18:00 Uhr)

(b) Unter Typisierung versteht man die Zuweisung von Objekten (z.B. Variablen) zu bestimmten Datentypen (z.B. Integern, Floats, Strings, etc.). Hierbei wird zwischen statischer und dynamischer Typisierung interschieden (in Python wird dynamische Typisierung verwendet). Bei statischer Typisierung muss einem Objekt schon bei seiner Definition ein fester Datentyp zugeordnet werden, der danach nicht mehr verändert werden kann. Bei dynamischer Typisierung ist dies nicht der Fall, hier müssen Objekte nicht direkt einem Datentyp zugeordnet werden und der Datentyp einer Variablen kann wären dem Programm verändert werden (z.B. ein Integer kann in einen Float oder einen String umgewandelt werden).

(Quelle: http://www.inztitut.de/blog/glossar/typisierung/; Datum: 22.10.17; Uhrzeit: 18:20 Uhr)

Marvin Glaser
4424114

Gruppe 9
Wilhelm Esser

1.2

(a) Die „sum()“ Funktion von in Python summiert alle in einer „*Iterable*“ angegebenen Elemente mit einem definierten Startwert „*start*“ (Standard = 0). Der Syntax ist hierbei: sum(iterable, start) oder einfach sum(iterable). Hierbei dürfen weder die *iterable*, noch *start* Strings enthalten.

(Quelle: https://docs.python.org/2/library/functions.html#sum; Datum: 22.10.17; Uhrzeit: 18:30 Uhr)

(b) TypeError:
Ausgegeben, wenn eine Operation oder eine Funktion auf ein Objekt angewendet wird, dass nicht unterstützt wird (Bsp.: „“.join(1)) oder zu viele Objekte angegeben sind (Bsp.: sum(list, list2, 0])

IndexError:
Ausgegeben, wenn der Index einer Sequenz aufgerufen wird, der nicht existiert
Bsp.:
list = [„a“, „b“, „c“];
b = a[3]
list[0], list[1] und list[2] existieren, aber list[3] nicht → IndexError

SyntaxError:
Ausgegeben, wenn ein Syntaxfehler (fehlerhafte Zusammensetzung von Zeichen innerhalb des Regelsystems der Programmiersprache) in dem ausgeführten Code vorliegt.
Bsp.:
print „hello world“

NameError:
Ausgegeben, wenn ein Objekt (Variable, Funktion, etc.) aufgerufen wird, dass nicht gefunden werden kann (z.B. nicht definiert ist).
Bsp.:
list = [1, 2, 3]
print(list2)

(Quelle: https://docs.python.org/3.6/library/exceptions.html#bltin-exceptions; Datum: 25.10.17; Uhrzeit: 20:55 Uhr)

Marvin Glaser
4424114

Gruppe 9
Wilhelm Esser

1.3

(a) Terminal: Das Terminalfenster (oder auch Konsole) ermöglicht die Bedienung des Computers mithilfe von Befehlen.

Python Shell: Dieses Fenster zeigt ein Programm das genutzt wird um andere Programme darin auszuführen (hier Python). Die Shell dient sozusagen als „Hülle“ für das ausgeführte Programm.

(b) Command Prompt (Windows 10 Terminal):

chdir (kurz cd): Zeigt Ordner an, in dem sich der User gerade befindet. Kann auch genutzt werden, um zwischen Orndern zu navigieren.

dir: Zeigt alle Ordner und Programme an, die sich in dem Ordner befinden, in dem sich der User gerade aufhält.

del: Wird verwendet um einen oder mehrere Daten zu löschen.

(Quelle: https://www.lifewire.com/list-of-command-prompt-commands-4092302; Datum: 25.10.17; Uhrzeit: 22:00)

Python Shell (IDLE, Windows 10):

input(„Text“): Gibt dem User „Text“ als Textprompt und ermöglicht die Eingabe von Informationen. Kann genutzt werden, um Variablen zu definieren.

type(o*bject*): Gibt den Typ von Objekt aus (z.B. Integer, Float, String, List, etc.)

abs(*object*): Nimmt als Argument ein Objekt das ein Integer, ein Float oder eine komplexe Zahl ist und gibt den absoluten Wert aus.

(Quelle: https://docs.python.org/3/library/functions.html; Datum: 25.10.17; Uhrzeit: 22:00 Uhr)

1.4

(a) Eingabe: Erfüllt nicht die Vorgabe eines Algorithmus. Es fehlt die Vorschrift, dass die Liste endlich sein muss.

Ausgabe: Erfüllt die Definition eines Algorithmus, da alle notwendigen Kriterien für einen Algorithmus erfüllt sein können (hier besonders: ausführbar, allgemeingültig, determiniert und Endlichkeit der Ausgabe)

Vorschrift: Kriterien erfüllt. Es gibt zu jedem Zeitpunkt der Ausführung nur eine Möglichkeit den Algorithmus fortzuführen (deterministisch) und der Algorithmus ist endlich (solange die Liste endlich ist; siehe Eingabe).

(b) Eingabe: Eine Liste von 8 Karten, mit den möglichen Varianten 7, 8, 9, 10, Dame, Bube, König und Ass. Jede Variante kann bis zu vier Mal, in jeweils unterschiedlichen Farben (Kreuz, Pik, Herz, Karo) angegeben werden. Keine der Karten in der Liste darf identisch sein.

Ausgabe: Eine Liste der eingegebenen Karten, sortiert nach Wertigkeit 0 (und dann aufsteigend nach 7, 8, 9), 2 (Bube), 3 (Dame), 4 (König), 10 (10) und 11 (Ass). Es findet keine Sortierung der Karten nach Farbe statt.

Vorschrift: Beginne mit dem ersten Element der Liste und prüfe immer das jeweils nächste Element. Prüfe für jedes Element der Liste, ob der Wert des vorherigen Elements kleiner, gleich oder größer ist. Ist der Wert des vorherigen Elements größer, dann tausche die Positionen der Elemente und prüfe dasselbe Element erneut. Führe dies aus, bis das letzte Element der Liste einen größeren oder gleichen Wert hat, als sein vorheriges Element. Dann beginne erneut am Anfang der Liste und prüfe, ob es sich bei dem Element um eine 7, 8 oder 9 handelt. Ist dies der Fall prüfe, ob das vorherige Element einen größeren Zahlenwert hat als das aktuelle. Ist dies der Fall, tausche die Positionen der beiden Elemente und prüfe dasselbe Element erneut. Prüfe dann das nächste Element auf dieselbe Weise. Brich diese Prüfung ab, sobald das untersuchte Element keine 7, 8 oder 9 ist.