

# PLÜ Sem2-1 1AHIF-POS, 2021-03-19, Gr. A

## Inhaltsverzeichnis

- [1. Aufgabe: Debugger](#)
- [2. Aufgabe: Zeichnen mit Zeichen](#)
- [3. Aufgabe: String-Methoden, Random, Konstante](#)

### Allgemeine Hinweise:

- Das Projekt wird am Desktop angelegt (Standardeinstellung!).
- Projektname ist der **Familienname** mit großem Anfangsbuchstaben, Rest klein (z.B.: **Mustermann**).
- Am Desktop der Prüfungsumgebung liegt die Javadoc-Dokumentation. Darin kann für Random oder String etc. nachgesehen werden.

## 1. Aufgabe: Debugger

Gegeben ist folgende Klasse:

```
public class A1a_Debugger
{
    public int nr = 17;

    public void Einstieg() {
        String txt = "Trick" + nr;
        String endTxt = work(txt);
        System.out.println(endTxt.substring(1,3));
    }

    private String work(String inTxt) {
        String workTxt = "";
        for (int i = 0; i < 3; i++) {
            workTxt += "." + inTxt.charAt(i);
            if (i >= inTxt.length()) {
                return workTxt + '-' + i;
            }
        }
        return workTxt + "_all";
    }
}
```

Setze einen **Breakpoint** in die erste Zeile der Methode **einstieg** (String txt = "Trick" + nr; ).

**ACHTUNG:** Es darf **KEIN weiterer Breakpoint** gesetzt werden! Dieser Breakpoint muss auch im zu erstellenden Screenshot sichtbar sein!

Kopiere die im Ordner PLÜ-1AHIF-2021-03-19 (am Desktop) liegende Datei `Debugger-Step-Liste.txt` (Inhalt siehe folgende Tabelle) in die `README.txt`-Datei im BlueJ-Projekt und fülle sie wie in den bereits ausgefüllten Einträgen gezeigt **für jeden Schritt** im Debugger bis zum Ende des Programms aus (die ersten 10 Schritte sind schon korrekt ausgefüllt, setze diese fort).

**VORSCHLAG:** Man könnte auch die oben genannte Datei `Debugger-Step-Liste.txt` mit dem Windows-Notepad (oder besser Notepad++) fertigstellen und am Schluss in die `README.txt`-Datei kopieren.

Die Tabelle sollte EXAKT stimmen und **exakt wie vorgegeben formatiert** sein, jeder Klick auf [Step] oder [Step Into] eine Zeile!

**WICHTIG:** Einschalten der Zeilennummer: Menü *Options/Preferences...* → Tab "Editor" → Checkbox "Display Line Numbers"

Variable, die aktuell nicht im Debugger-Fenster sichtbar sind, erhalten Leerzeichen.  
Strings sind in "" einzuschließen (falls null: null schreiben).

Die erste Zeile enthält als Spaltenbeschriftungen:

- die Zeilennummer der grünen Markierung
- zu drückender Button
- Info bei markanten Zeilen (dekl ... Variablen-Deklaration)
- die 6 Variablennamen

Hier die ersten 10 Einträge als Muster, die gleich bestehen bleiben können:

ZEILE	Button	INFO	nr	txt	inTxt	workTxt	i	endTxt
6	STEP	dekl	17					
7	STEP-INTO	Methode	17	"Trick17"				
12	STEP	dekl	17		"Trick17"			

13	STEP	for	17	"Trick17"	"	
14	STEP		17	"Trick17"	"	0
15	STEP	if	17	"Trick17"	".T"	0
13	STEP	for	17	"Trick17"	".T"	0
14	STEP		17	"Trick17"	".T"	1
15	STEP	if	17	"Trick17"	".T.r"	1
13	STEP	for	17	"Trick17"	".T.r"	1

Erzeuge einen **Screenshot** (Rechteckiger Bereich) mit den unten genannten **2 vollständig sichtbaren Fenstern**, wenn die **Ausführungspositions-Markierung auf Zeile 19: `return workTxt + "_all";`** steht:

- Editor mit sichtbarem Breakpoint und sichtbarer grüner Ausführungspositions-Markierung
- Debugger-Fenster

und kopiere die Screenshot-Datei in das BlueJ-Projektverzeichnis.

## 2. Aufgabe: Zeichnen mit Zeichen

Erstelle eine Klasse `A2a_Zeichnen` mit Methode `void zeichne(int hoehe) // Höhe min 1, max 20`, die folgendes Muster erzeugt (bei `hoehe=3`):

```

\
 |
 |

```

Hilfsmethoden erlaubt. Für den Backslash muss geschrieben werden: `'\\'`.

## 3. Aufgabe: String-Methoden, Random, Konstante

Erstelle eine Klasse `A3_TextUndZufall`, die mittels Zufallszahl einen der Namen ausgibt.

Es gibt die **Konstante** `LISTE_NAMEN` mit Inhalt `"Personen: Ada Adi Edi Evi Ida Udo Ute"`. Diese ist (bis auf das erste Wort) die Quelle der verfügbaren Namen.

**TIPP:** String-Methoden für die Konstante werden wie üblich aufgerufen: `LISTE_NAMEN.charAt(...)`.

**ACHTUNG:** die Logik muss auch funktionieren, wenn in der Konstanten das Wort `"Personen:"` durch EIN anderes Wort ersetzt wird und auch andere Namen enthalten sind! ( `' : '` gilt als Teil des Wortes)

Es gibt folgende `public` Methoden:

- **`String zufallsName()`** ... ist die "Hauptmethode", die aus der Konstanten `LISTE_NAMEN` zufällig (mit Klasse `java.util.Random`) einen Namen zurückliefert (**Achtung: erstes Wort ist KEIN Name!**). Diese nutzt intern die nachfolgenden Methoden zur Erreichung des Ziels:
  - Zuerst wird eine Zufallszahl innerhalb der benötigten Grenzen (`anzahlLeerz()` !!) generiert (**Erinnerung: Import nicht vergessen; erst Zufallszahlen-Generator instanzieren, damit `next...`**)
  - damit wird Methode `extrahiereNameMitNr(...)` aufgerufen, die intern `findePositionLeerzNr(...)` nutzt.
- **`int anzahlLeerz()`** ... liefert die Anzahl der in der Konstanten `LISTE_NAMEN` enthaltenen Leerzeichen (und damit auch die Anzahl der gültigen Namen, da das erste Wort kein Name ist)
- **`int findePositionLeerzNr(int leerzNr)`** ... geht mittels String-Methode `charAt(...)` zeichenweise die String-Konstante durch und liefert die **Position** des N-ten (definiert durch den Parameter) Leerzeichens aus. Falls kein Leerzeichen gefunden wird, wird `-1` geliefert.
- **`String extrahiereNameMitNr(int namensNr)`** ... liefert den N-ten **Namen** aus der String-Konstanten zurück. Falls zu hohe Nummer übergeben, wird `null` geliefert. Intern wird zuerst `findePositionLeerzNr(...)` aufgerufen, das (nach Erhöhung um 1) die Startposition liefert. Das Ende kann durch nochmaligen Aufruf mit um eins erhöhter Nummer gefunden werden. Das Wort selbst kann entweder mit `substring(...)` oder wieder mit `charAt(...)` extrahiert werden.

**HINWEIS:** Wenn eine Methode nicht funktioniert, kann sie umbenannt UND wenn nötig auskommentiert werden auf z.B. `anzahlLeerz_PROBLEM()` und eine "Fake"-Implementation geschrieben werden, um den Rest testen zu können:

`anzahlLeerz_FAKE() { return 7; }`. Das wirkt sich gut auf die Note aus! (Wichtig ist die Kenntlichmachung als "Fake")

Last updated 2021-03-18 20:48:56 +0100