# Blatt 4

## Aufgabe 1

Der Stack wird benötigt, um nach dem Ausführen von Unterprogrammen wieder zum Aufrufer zurückzukehren. Außerdem werden dort die Parameter eines Funktionsaufrufs abgelegt.

Die Befehle sind pushl (einen Wert auf den Stack legen) und popl (den obersten Wert vom Stack holen und löschen). Movl %esp [Zielregister] liest einen Wert vom Stack ohne ihn zu löschen.

## Aufgabe 2

* Zuerst werden die Werte 2 und 4 auf den Stack geschoben (1)
* Dann wird die Funktion tuwas aufgerufen (2)
* Der Stackpointer wird wieder zurückgesetzt (der durch (1) manipuliert worden ist) (3)
* Sichern des alten Basepointers auf den Stack (4.1)
* Überschreiben des Basepointers mit dem Stackpointer (4.2)
* Der Stackpointer wird um zwei Register nach unten bewegt um Platz für lokale Variablen zu reservieren (5)
* Der Stackpointer wird mit dem Basepointer überschrieben (6.1)
* Der alte Basepointer wird wiederhergestellt, indem der zuvor gesicherte BP in den aktuellen geschrieben wird (6.2)
* Vom Stack wird gepopt und in den eip geschrieben. Damit wird die Rücksprungsadresse in den Instructionpointer geschrieben. (7)

## Aufgabe 3

* Die Reihenfolge ist umgedreht. Bei der Funktion foo(a, b, c) liegen im Speicher von oben nach unten c, b, a.
* Bei a: Wert wird in eax geschrieben. Bei b: Pointer wird in eax geschrieben.
* Der Aufrufer muss sich um die Sicherung kümmern, da der Aufgerufene alle Register überschreiben / löschen kann.
* Annahme: Unter Korrektur des Stacks ist das Zurücksetzen des Stackpointers (esp) gemeint. Dafür ist wieder der Aufrufer zuständig.