a) Schreiben Sie ein Programm für die **Stackmaschine**, das die Berechnung

`(a+b) x c / d`

durchführt (und dazu die Befehle `push`, `pop`, `add`, `mul`, `div` verwendet). Die Inhalte von `a`, `b`, `c`, `d` liegen in den Speicherzellen `0x1000` bis `0x1003`, das Ergebnis soll in `0x1004` liegen.

```
push 0x1000  // a
push 0x1001  // b
add
push 0x1002  // c
mul
push 0x1003  // d
div
pop 0x1004   // Ergebnis
```

Falls div die Operanden in anderer Reihenfolge erwartet, ist folgende Lösung möglich:

```
push 0x1003  // d: Stack = [d]
push 0x1000 // a:  Stack = [d,a]
push 0x1001 // b:  Stack = [d,a,b]
add                // a+b:       = [d, a+b]
push 0x1002 // c                 = [d, a+b, c]
mul                // (a+b)*c    = [d, (a+b)c]
div                //              [ d / ((a+b)c) ]
pop 0x1004  //                   [ ]
```

b) Geben Sie für die beiden Befehle PUSH und ADD an, ob es sich jeweils um einen 0-, 1-, 2- oder 3-Adress-Befehl handelt.

push: 1- Adress-Bef.
add: 0- Adress-Bef.

Bsp. für viele Adressen:
mov eax, [ebx + 4*ecx + 0x1234] <— 5-Adress-Befehl
(auch „4“ ist eine Adresse!)