Lower Alpha

interaktive α-Notation

Sirko Höer
s6sihoee@uni-bonn.de
Maximilian S.
s6mnsimo@uni-bonn.de
Jan Müller
muellerj@informatik.uni-bonn.de

Motivation & Ziele

- Interaktive Umgebung für α-Notation
- Nähe zu Übungsaufgaben & Vorlesung

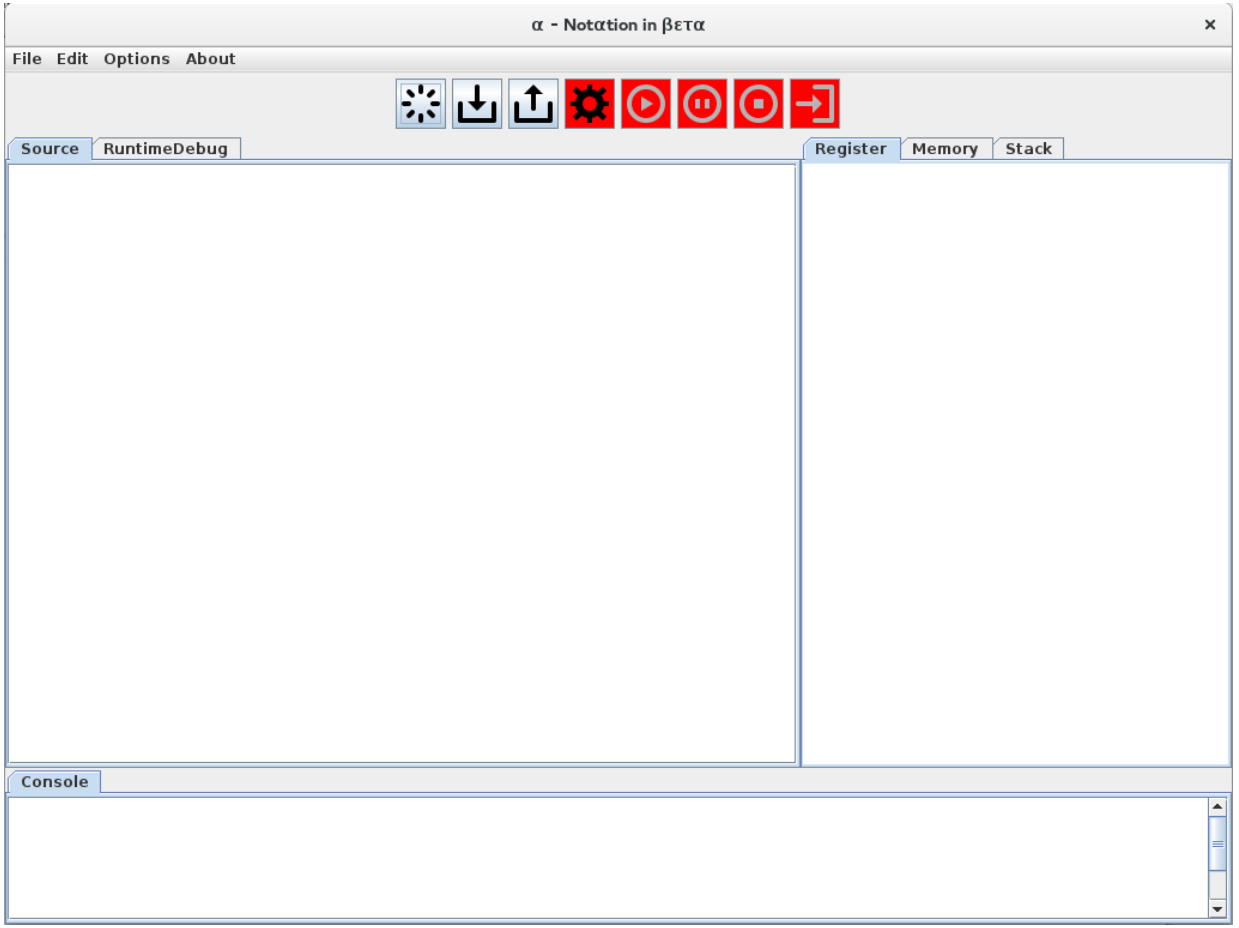
Einschränkungen

- Beschränkte verschachtelte Speicherzugriffe
  - Höchsten ρ(ρ(<Wert>))
- Erweitertes 3-Adress-Format
  - Selbst auf zulässige Befehle achten
- Uninitialisierte Register, Speicherzellen beenden das Programm

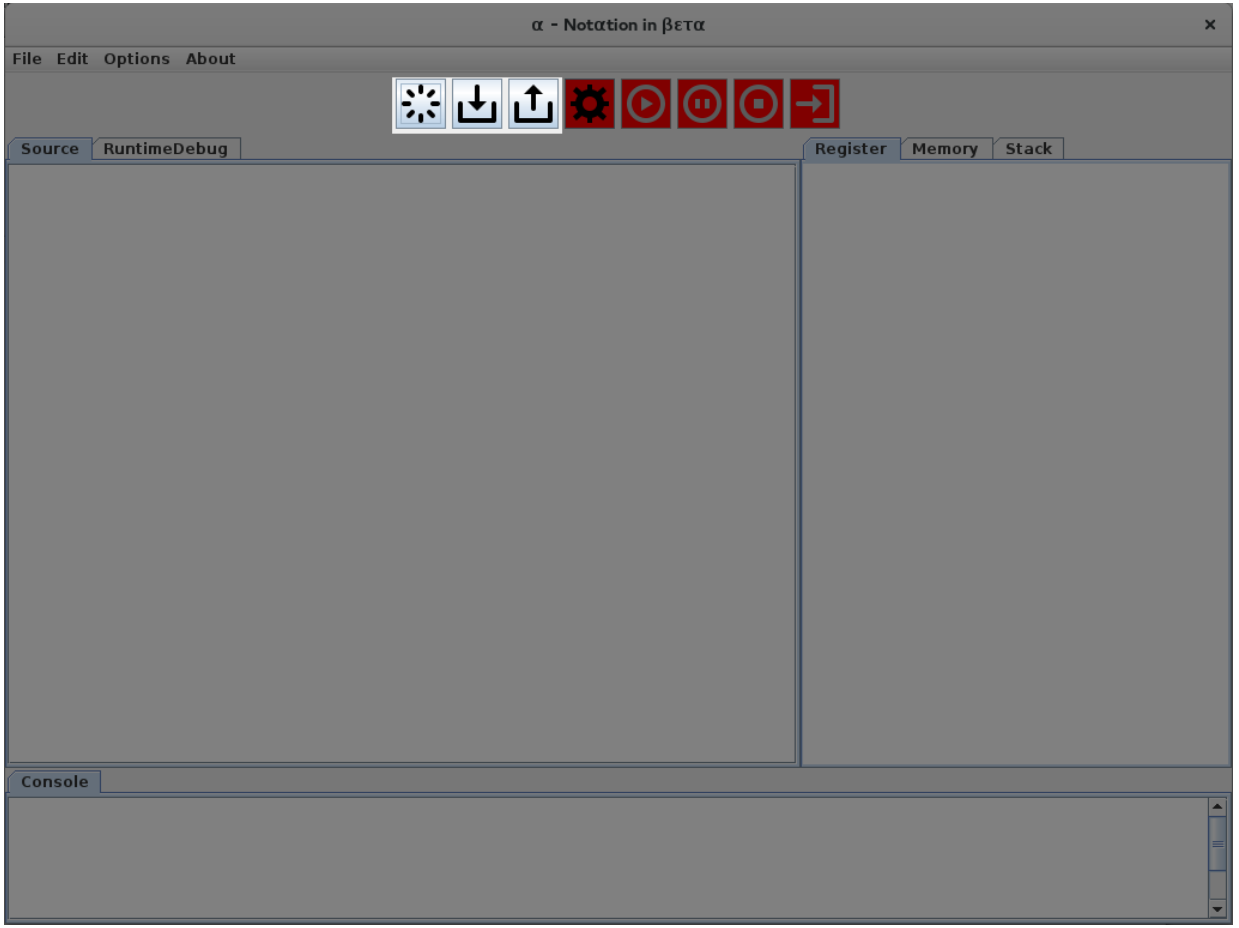
Spezielle Funktionen

- Das Label „main:“ makiert den Einstiegspunkt
  - Ist aber optional
- stackbasierte Rücksprungadressen
  - Alle Programme müssen mit „return“ enden

Userinterface



Projekte verwalten

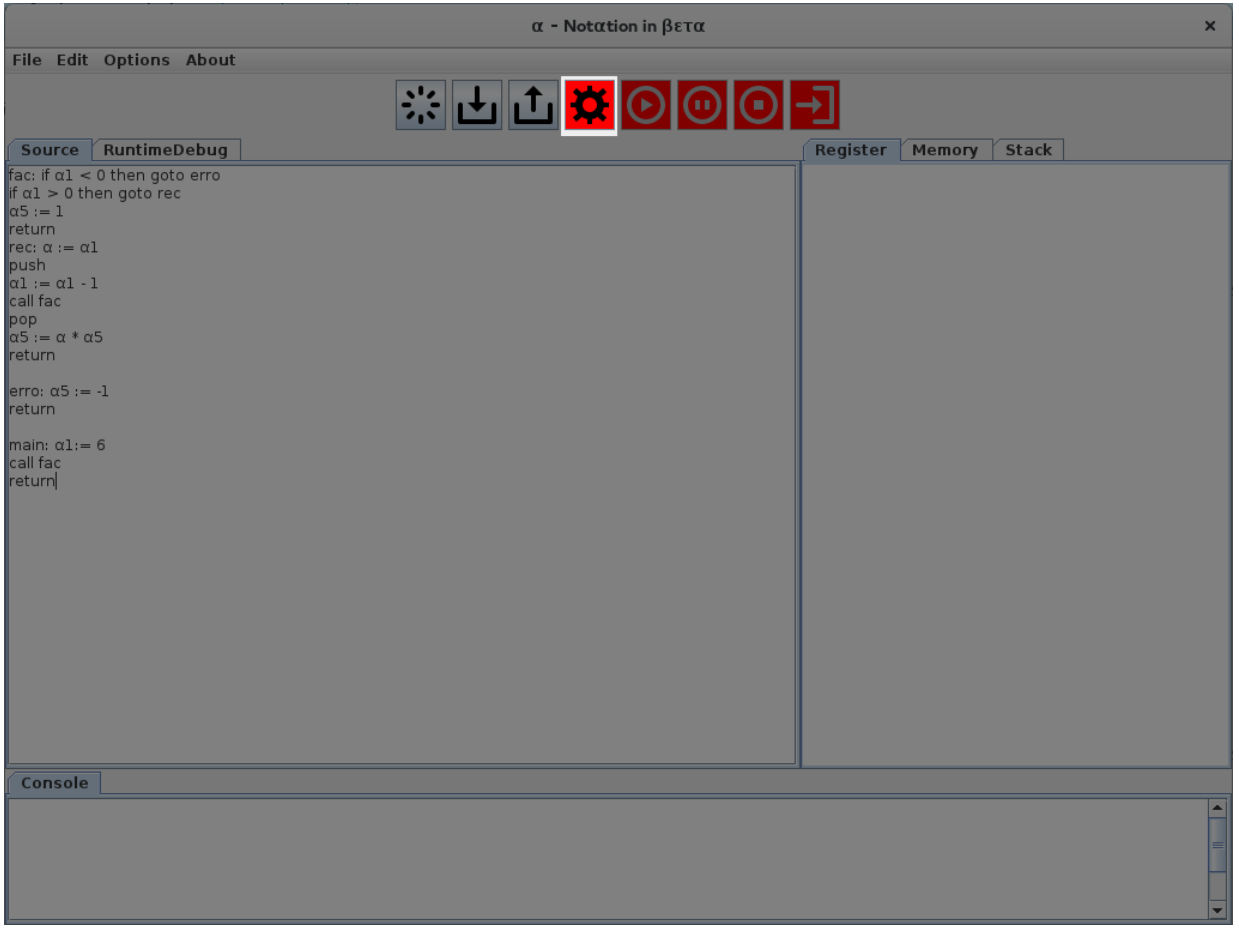


Schaltflächen: Neues Projekt öffnen, Projekt speichern, Projekt öffnen
(Projekte sind UTF-8 kodierte Textdateien)

Shortcuts

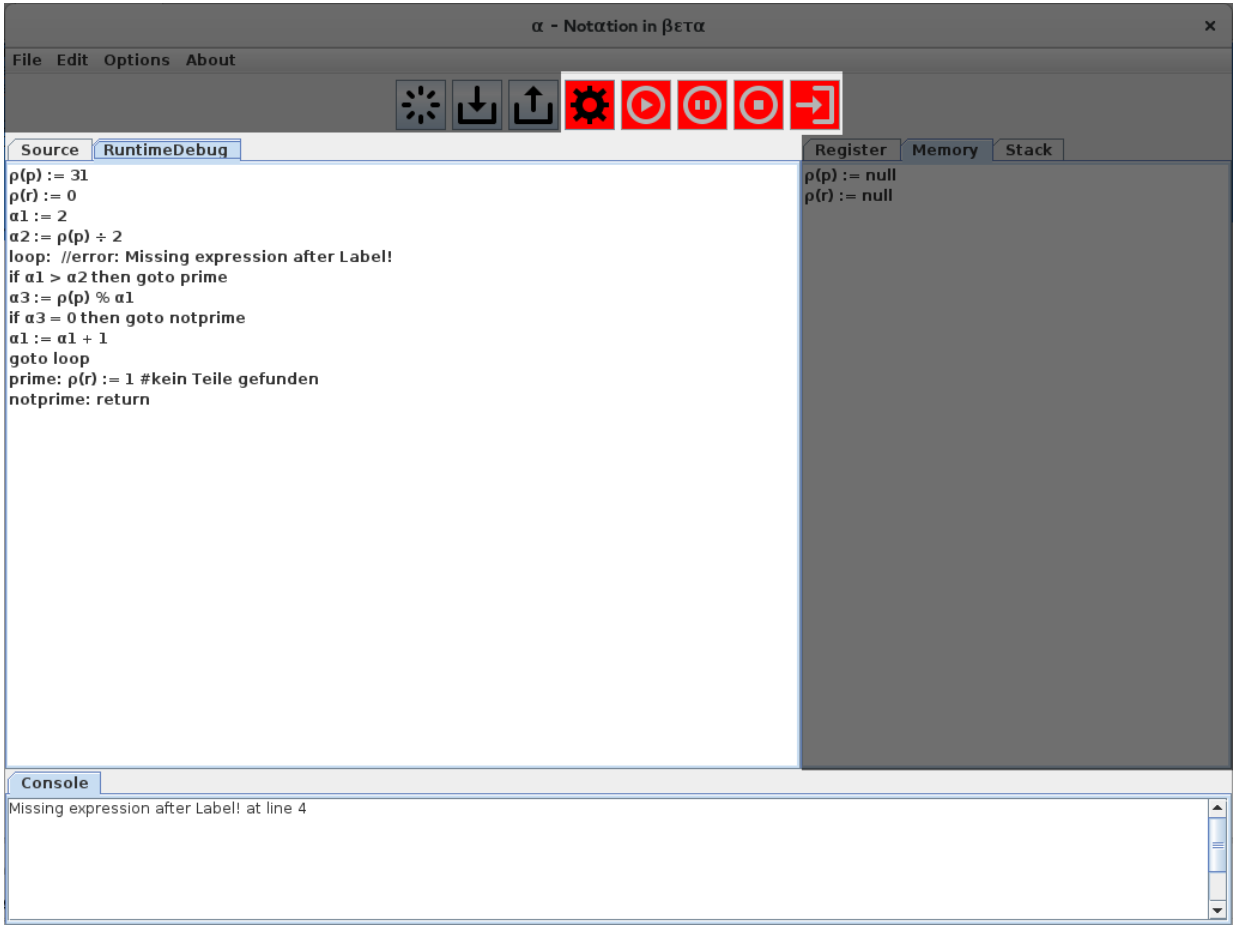
Shortcut	Ergebnis
alpha, a;	α
rho, r;	ρ
a,	α :=
r,	ρ() :=
if:	if var1 comp var2 then goto label
if=	if var1 = var2 then goto label
if<	if var1 < var2 then goto label
if>	if var1 > var2 then goto label

Programm erstellen



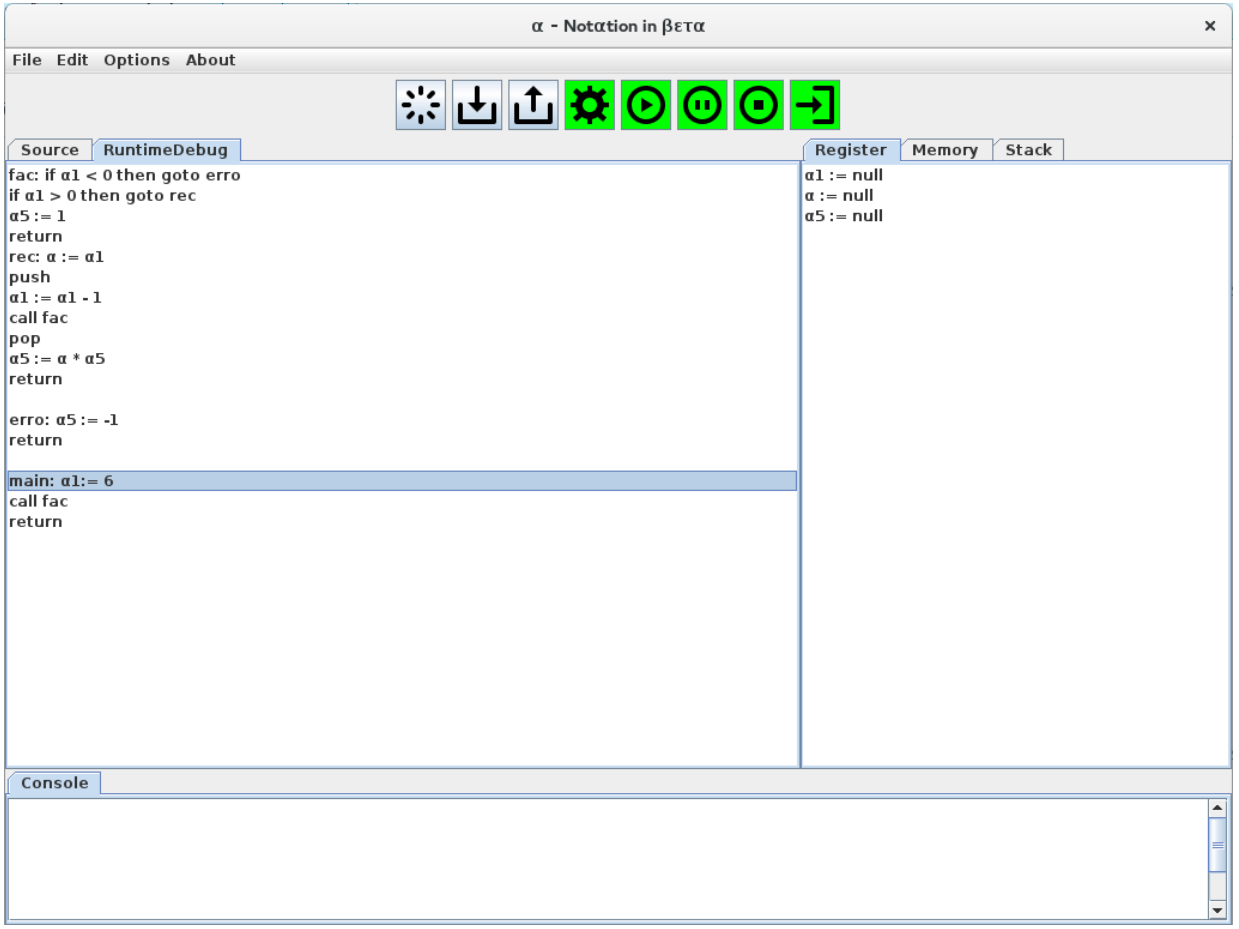
Die „Zahnrad“-Schaltfläche erstellt das geschriebene Program

Compilierfehler



Enthält das Programm einen Fehler, bleiben alle Schaltflächen rot; in der „RuntimeDebug“-Ansicht und Konsole wird der Fehler ausgegeben.

Compilieren

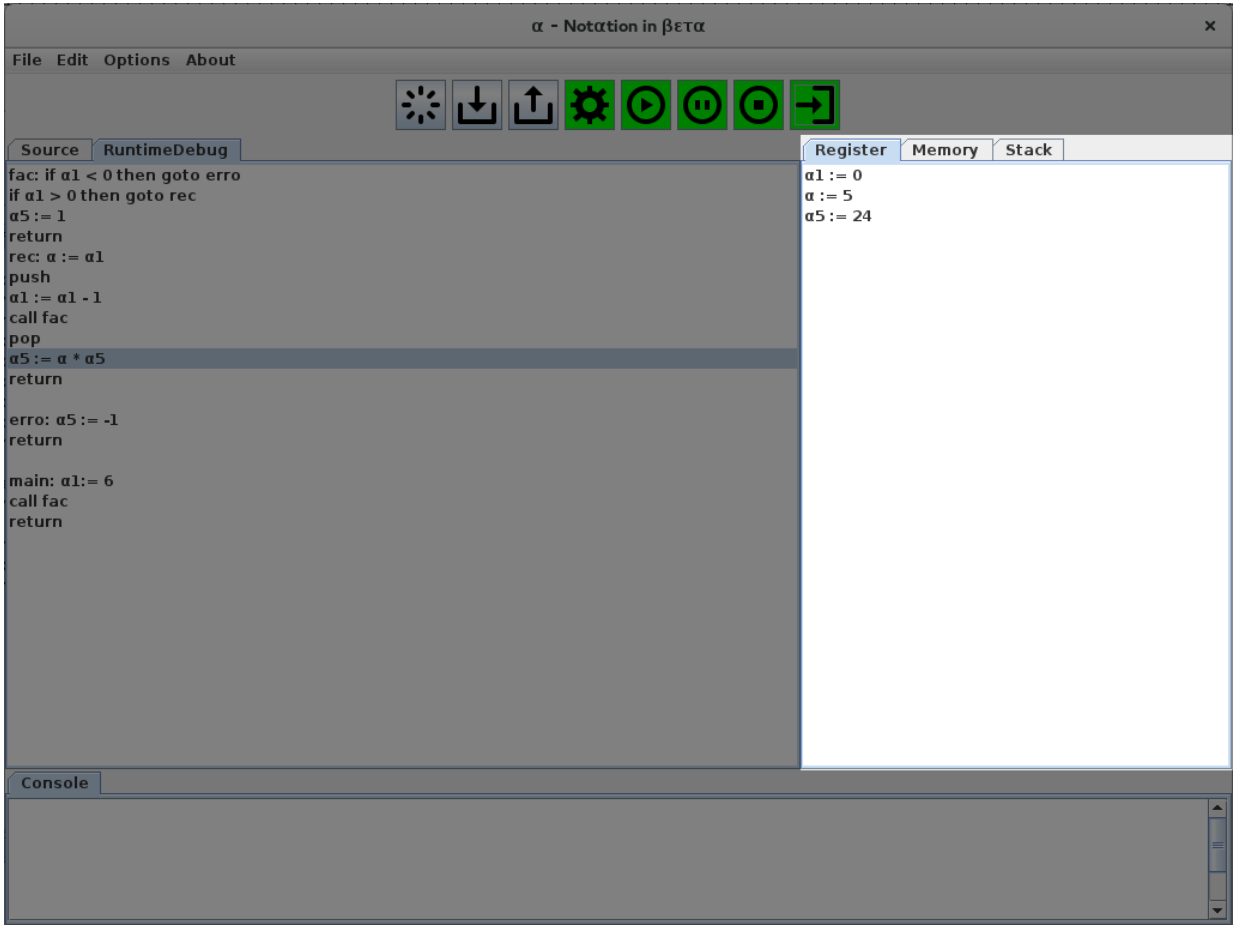


wurde das Programm erfolgreich kompiliert, sind alle Schaltflächen grün und das Programm kann gestartet werden

Ausführen

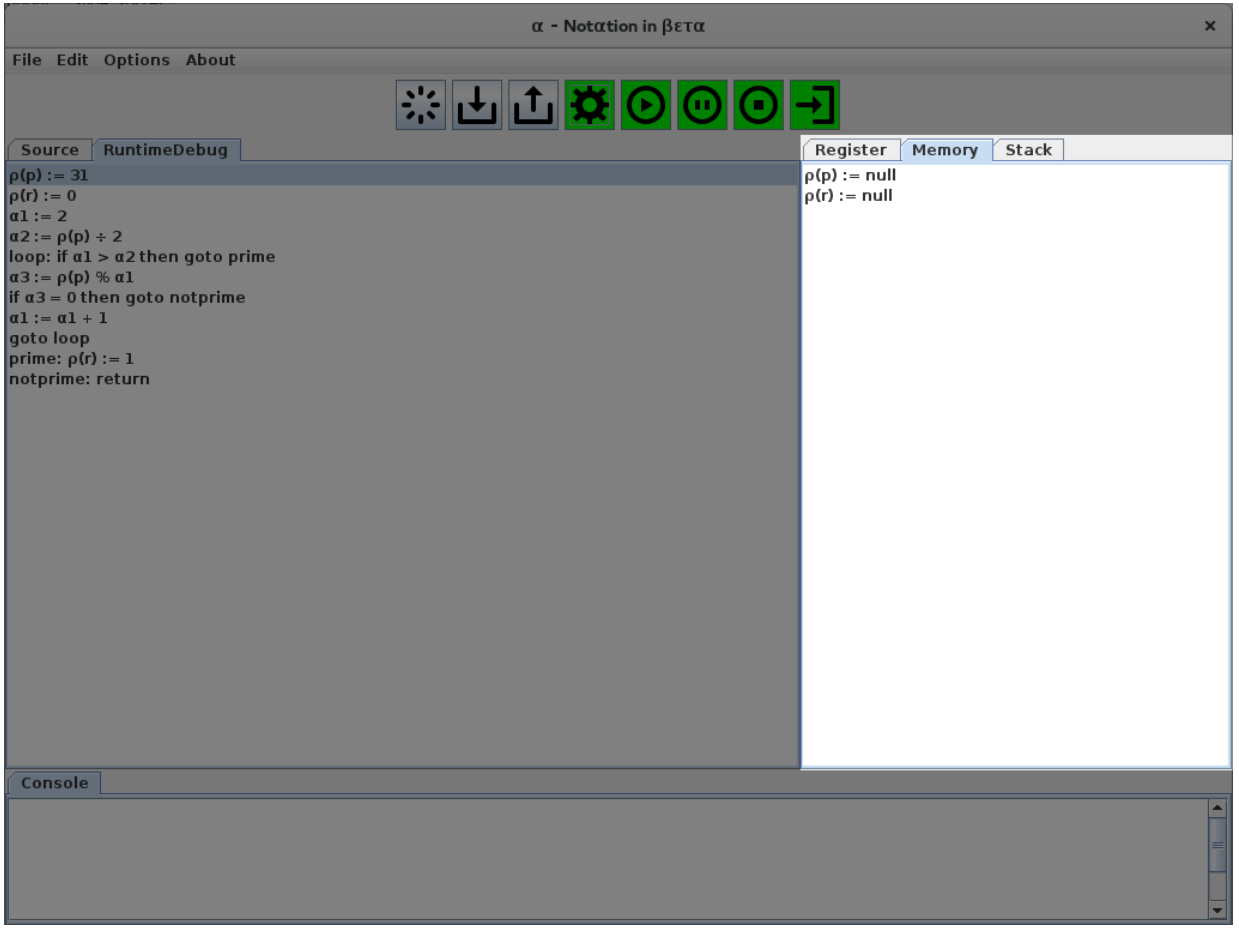
Entweder automatisches Durchlaufen oder zweilenweise durchgehen.

Debuggen



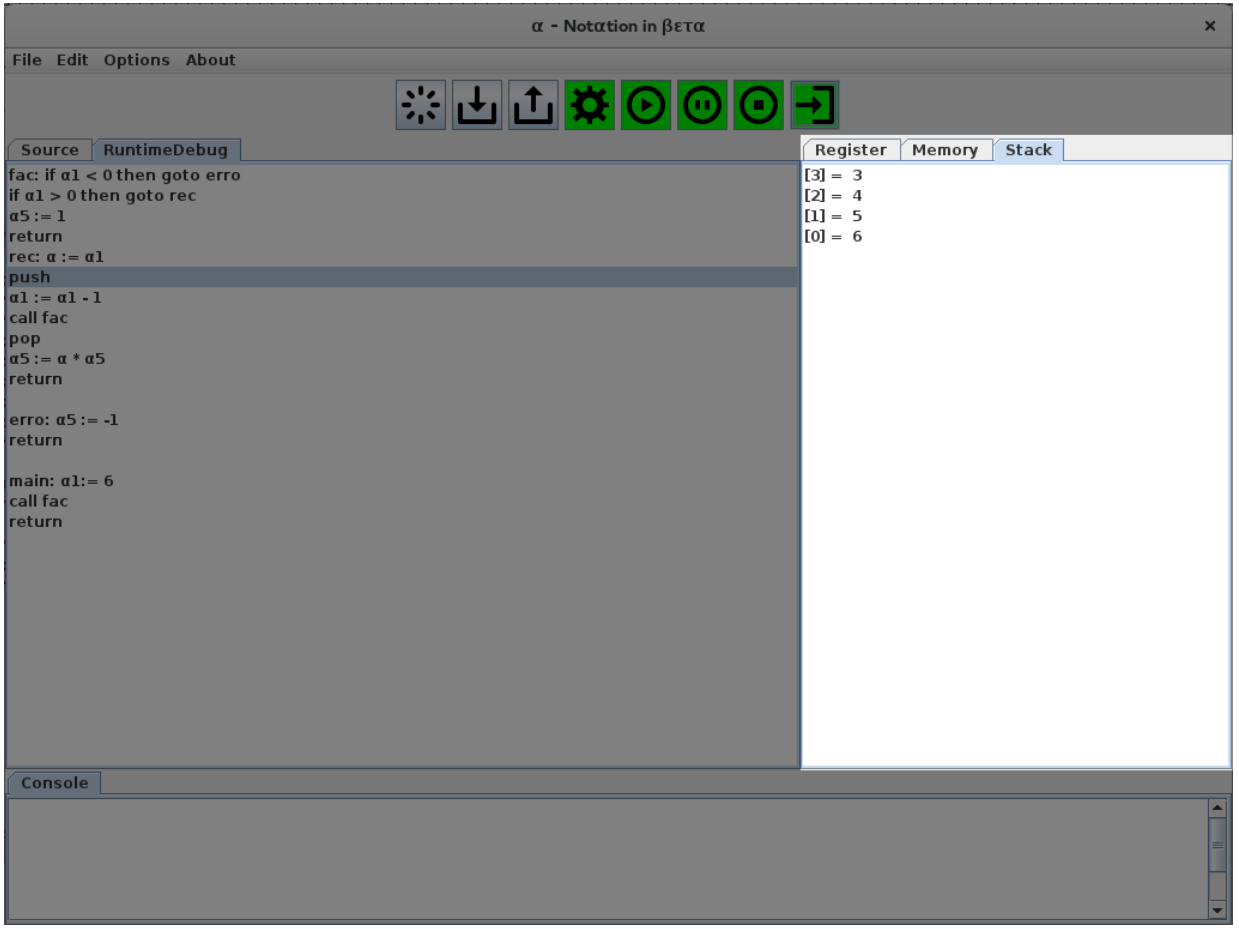
Register, Speicher, Stack werden automatisch aktualisiert (auch bei automatischem Ausführen)

Debuggen - Speicher



Nicht initialisierte Werte werden mit „null“ gekennzeichnet (auch bei Regsitern)

Debuggen - Stack



Das obestere Element, ist das zuletzt auf den Stack gelegte Element

Demos

GitHub-Link

https://github.com/SirkoHoeer/LowerAlpha.git

YEEE It compiles, we ship it!!! ;-)