

Programmieren 1 - WS 2020/21

Übungsblatt 4

Aufgabe 1: Skript

a) Was ist die Aufgabe von constructor-Funktionen und was die von accessor-Funktionen?

Bei einer constructor-Funktionen werden Daten in einem Array mit Symbolen als Attributnamen definiert, um bestimmte Werte zu erstellen und initialisieren. Besonders bei der Erstellung von Punkt-Werten kann eine solche constructor-Funktion behilflich sein.

Mit Hilfe der accessor-Funktion lässt es sich mit nur geringen Datenverbrauch auf die constructor-Funktion zugreifen: das Element aus dem Array wird direkt aufgerufen (z.B. `f .0`, um auf das erste Element in der Array zuzugreifen).

b) Welchen Vorteil bietet `datadef` im Zusammenhang mit Variant Data?

Mithilfe von `datadef` kann man neue Datentypen definieren.
`datadef` erstellt constructor und accessor funktionen von sich aus.
So entfällt das händische definieren von `abc?`, `abc-x`, `abc-y` bei erstellen einer data mit Namen "abc". Außerdem erstellt `datadef` automatisch Testfunktionen.

c) Was versteht man unter Induktionshypothese im Zusammenhang mit rekursiven Funktionen?

Gegeben sei eine conditional function, die Grenzfälle sind bereits definiert, und es geht nun zu einer Stelle (laut script ein "leap of faith"), einem sog. Point of no return.

Man ruft die Funktion selbst auf und nimmt an (Induktionshypothese), dass die Funktion irgendwann eine Grenze erreicht.

Würde man dies nicht annehmen, ginge man davon aus, dass die Funktion hier immer wieder ankommt, sich wieder aufruft und es so endlos weitergeht.

d) Was war Ihnen beim Lesen der Kapitel 7-9 unklar? Wenn nichts unklar war, welcher Aspekt war für Sie am interessantesten?

Dass man aus selbstdefinierten Data-types einzelne Teile rausziehen, und verwenden kann ist recht Interessant (vergl 9.2 Exercises).

Man kann eine data-definition für Personen erstellen, mit Name, Geburts- und Sterbejahr und dann das Durchschnittsalter ausgeben lassen.

Scheint Recht komplex aber interessant.

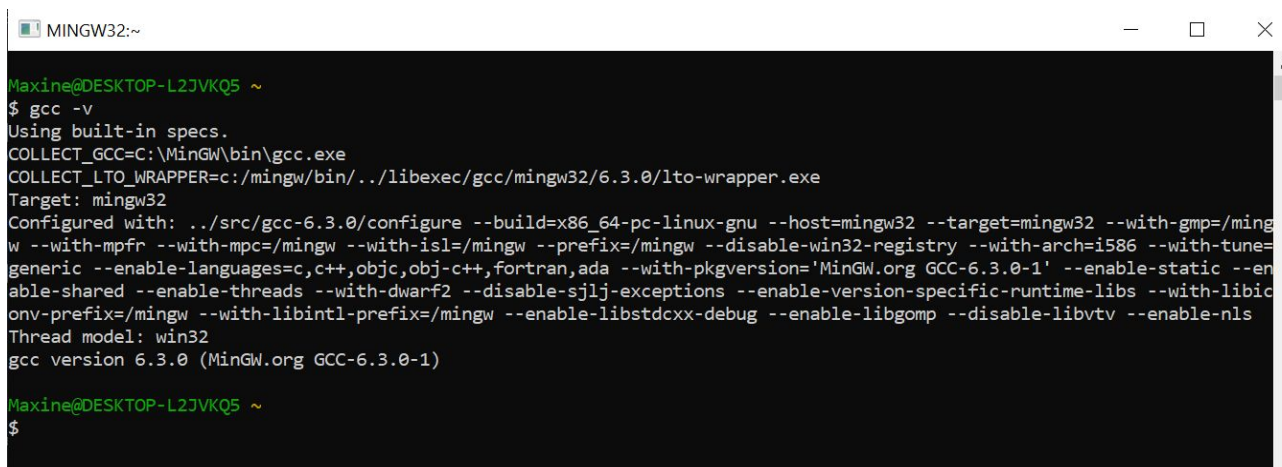
Aufgabe 2

Siehe Datei "Aufg_2"

Aufgabe 3
Siehe Datei "Aufg_3"

Aufgabe 4: C-Compiler installieren

a) gcc -v



```
MINGW32:~
Maxine@DESKTOP-L2JVKQ5 ~
$ gcc -v
Using built-in specs.
COLLECT_GCC=C:\MinGW\bin\gcc.exe
COLLECT_LTO_WRAPPER=c:/mingw/bin/./libexec/gcc/mingw32/6.3.0/lto-wrapper.exe
Target: mingw32
Configured with: ../src/gcc-6.3.0/configure --build=x86_64-pc-linux-gnu --host=mingw32 --target=mingw32 --with-gmp=/mingw
--with-mpfr --with-mpc=/mingw --with-isl=/mingw --prefix=/mingw --disable-win32-registry --with-arch=i586 --with-tune=
generic --enable-languages=c,c++,objc,obj-c++,fortran,ada --with-pkgversion='MinGW.org GCC-6.3.0-1' --enable-static --en
able-shared --enable-threads --with-dwarf2 --disable-sjlj-exceptions --enable-version-specific-runtime-libs --with-libic
onv-prefix=/mingw --with-libintl-prefix=/mingw --enable-libstdcxx-debug --enable-libgomp --disable-libvtv --enable-nls
Thread model: win32
gcc version 6.3.0 (MinGW.org GCC-6.3.0-1)

Maxine@DESKTOP-L2JVKQ5 ~
$
```

Stephan Jokiel:
<C:\Users\Steppo\Projekte\c>gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=C:/TDM-GCC-64/bin/./libexec/gcc/x86_64-w64-mingw32/9.2.0
/lto-wrapper.exe
Target: x86_64-w64-mingw32
Configured with: ../../src/gcc-git-9.2.0/configure --build=x86_64-w64-mingw32
--enable-targets=all --enable-languages=ada,c,c++,fortran,lto,objc,obj-c++ --enable-libgomp
--enable-lto --enable-graphite --enable-cxx-flags=-DWINPTHREAD_STATIC
--disable-build-with-cxx --disable-build-poststage1-with-cxx --enable-libstdcxx-debug
--enable-threads=posix --enable-version-specific-runtime-libs --enable-fully-dynamic-string
--enable-libstdcxx-threads --enable-libstdcxx-time --with-gnu-ld --disable-werror --disable-nls
--disable-win32-registry --enable-large-address-aware --disable-rpath --disable-symvers
--prefix=/mingw64tdm --with-local-prefix=/mingw64tdm --with-pkgversion=tdm64-1
--with-bugurl=http://tdm-gcc.tdragon.net/bugs
Thread model: posix
gcc version 9.2.0 (tdm64-1)

b)



```
MINGW32/c/Uni-Projekt

Maxine@DESKTOP-L2JVKQ5 ~
$ cd C:\Uni-Projekt

Maxine@DESKTOP-L2JVKQ5 /c/Uni-Projekt
$ gcc hello.c -o hello.exe

Maxine@DESKTOP-L2JVKQ5 /c/Uni-Projekt
$ hello.exe
hello, world

Maxine@DESKTOP-L2JVKQ5 /c/Uni-Projekt
$
```

Aufgabe 4b und 4c in der Datei "hello.c" bearbeitet

d)

mkdir Der Befehl "mkdir" erstellt einen neuen Ordner (directory) im aktuell ausgewählten dir.

ls Der Befehl "ls" listet alle Dateien und Ordner des aktuellen directorys in unix-systemen auf. ggf. ist dies "dir" unter windows.

cd Der Befehl "cd" wird benutzt, um das directory zu verändern "change directory". so kann man beispielsweise mit "cd.." einen Ordner zurückgehen und mit "cd Users" beispielsweise von "C:\\" nach "C:\Users" gelangen.

e)

1.

```
C:\Users\Steppo\Projekte\c\Assignment 4>tar -xf task4.zip

C:\Users\Steppo\Projekte\c\Assignment 4>dir
Volume in Laufwerk C: hat keine Bezeichnung.
Volumeserienummer

Verzeichnis von C:\Users\Steppo\Projekte\c\Assignment 4

06.11.2020  14:03    <DIR>        .
06.11.2020  14:03    <DIR>        ..
03.11.2020  14:36    <DIR>        folder
03.11.2020  14:38                1.184 task4.zip
                1 Datei(en),          1.184 Bytes
                3 Verzeichnis(se), 14.285.193.216 Bytes frei
```

Es ist im richtigen Verzeichnis entpackt.

In das Verzeichnis folder/A01 kommt man entweder über:

```
C:\Users\Steppo\Projekte\c\Assignment 4>cd folder\A01

C:\Users\Steppo\Projekte\c\Assignment 4\folder\A01>
```

oder einzeln:

```
C:\Users\Steppo\Projekte\c\Assignment 4>cd folder

C:\Users\Steppo\Projekte\c\Assignment 4\folder>cd A01

C:\Users\Steppo\Projekte\c\Assignment 4\folder\A01>
```

2.

```
C:\Users\Steppo\Projekte\c\Assignment 4\folder\A01>cd..

C:\Users\Steppo\Projekte\c\Assignment 4\folder>cd A02

C:\Users\Steppo\Projekte\c\Assignment 4\folder\A02>
```

Bzw.

```
C:\Users\Steppo\Projekte\c\Assignment 4\folder\A01>cd ../A02

C:\Users\Steppo\Projekte\c\Assignment 4\folder\A02>
```

3.

Man gelangt in den Ordner A01.

```
C:\Users\Steppo\Projekte\c\Assignment 4>cd folder/../folder/A01

C:\Users\Steppo\Projekte\c\Assignment 4\folder\A01>
```

4.

Wenn ls -al statt ls aufruft wird, bekommt man eine detaillierte Übersicht der Dateien des Ordners, in dem man sich gerade befindet.

(gleich quasi "dir" unter windows)

```
Maxine@DESKTOP-L2JVKQ5 /c/Uni-Projekt
$ ls -al
sh: ls-al: command not found

Maxine@DESKTOP-L2JVKQ5 /c/Uni-Projekt
$ ls -al
total 51
drwxr-xr-x  4 Maxine Administratoren  0 Nov 10 17:32 .
drwxr-xr-x 25 Maxine Administratoren 8192 Nov 10 16:42 ..
drwxr-xr-x  5 Maxine Administratoren  0 Nov  3 14:36 folder
-rw-r--r--  1 Maxine Administratoren  81 Nov 10 16:51 hello.c
-rwxr-xr-x  1 Maxine Administratoren 40764 Nov 10 16:51 hello.exe
-rw-r--r--  1 Maxine Administratoren 1184 Nov 10 17:28 task4.zip
drwxr-xr-x  2 Maxine Administratoren  0 Nov 10 16:45 test

Maxine@DESKTOP-L2JVKQ5 /c/Uni-Projekt
$
```