Aufgabe 2: ADT Liste

In dieser Aufgabe soll die ADT Liste auf unterschieldiche Arten in Erlang/OTP implementiert werden.

Aufgabenstellung

Eine **ADT Liste** ist zu implementieren: als einfach verkette Liste (Datei: **liste.erl**):

create: ∅ → list
isEmpty: list → bool
insert: list × pos × elem → list
delete: list × pos → list
find: list × elem → pos
retrieve: list × pos → elem
concat: list × list → list

Als erste Anwendungen sind ein **ADT stack** auf der ADT Liste zu implementieren (Datei **stack.erl**):

createS: ∅ → stack
push: stack × elem → stack
pop: stack → stack
top: stack → elem
isEmptyS: stack → bool

und eine auf dieser ADT stack implementierter **ADT queue**, die mit zwei expliziten Stacks (in-Stack; out-Stack: siehe Vorlesung) arbeitet (Datei **schlange.erl**):

createQ: ∅ → queue
front : queue → elem (Selektor)
enqueue : queue × elem → queue
dequeue : queue → queue (Mutator)
isEmptyQ : queue → bool

Als Vorbereitung für Aufgabe drei ist ein **ADT array** auf der ADT Liste zu implementieren (Datei **array.erl**):

initA: ∅ → array
setA: array × pos × elem → array
getA: array × pos → elem
lengthA: array → pos

Weitere Vorgaben:

- Die Liste beginnt bei Position 1 und das Array bei Position 0.
- Die Liste arbeitet nicht destruktiv, d.h. wird ein Element an einer vorhandenen Position eingefügt, wird das dort stehende Element um eine Position verschoben.
- Das Array arbeitet destruktiv, d.h. wird ein Element an einer vorhandenen Position eingefügt, wird das dort stehende Element überschrieben.
- Die Länge des Arrays wird bestimmt durch die bis zur aktuellen Abfrage größten vorhandenen und explizit beschriebenen Position im array.
- Das Array ist mit 0 initialisiert, d.h. greift man auf eine bsiher noch nicht beschriebene Position im Array zu erhält man 0 als Wert.
- Das Array hat keine Größenbeschränkung, d.h. bei der Initialisierung wird keine Größe vorgegeben.

Geeignete **Tests**, insbesondere für die queue sind zu implementieren. Die ADT hat als Minimum die in Definition 2.5 des Skripts festgelegten Schnittstellen anzubieten. Beachten Sie: Ihre ADT wird ggf. für die Praktikumsaufgabe drei mit einer der Implementierung einer anderen Gruppe ausgetauscht.

Abnahme

Da die Aufgaben des ADP sehr frühzeitig im WWW zur Verfügung stehen, wird die Abnahme stark auf eine **vorbereitende Arbeit** aufgebaut.

Bis Sonntag Abend vor Ihrem Praktikumstermin ist eine erste Skizze der Aufgabe als *.pdf Dokument (Dokumentationskopf nicht vergessen!) mir per E-Mail zuzusenden. Ggf. können offene Fragen mit gesendet werden. Die Skizze **muss** grob beschreiben, wie Sie sich die Realisierung denken. Als erfolgreich wird eine Skizze bewertet, wenn Ihre Kenntnisse bzgl. der gestellten Aufgabe eine erfolgreiche Teilnahme an dem Praktikumstermin in Aussicht stellen.

Am Tag des Praktikums findet eine Befragung von Teams statt. Die **Befragung muss erfolgreich absolviert werden**, um weiter am Praktikum teilnehmen zu können. Ist die Befragung nicht erfolgreich, gilt die Aufgabe als nicht erfolgreich bearbeitet. Als erfolgreich wird die Befragung bewertet, wenn Ihre Kenntnisse eine erfolgreiche Teilnahme an dem Praktikumstermin in Aussicht stellen. Bei der Befragung handelt es sich nicht um die Abnahme.

Abgabe: Unmittelbar am Ende des Praktikums, spätestens bis 19:00 Uhr am selben Tag, ist von allen Teams der Code abzugeben. Zu dem Code gehören die Sourcedateien, die ggf. erzeugten *.log etc. Dateien, die während der Tests erzeugt wurden, und eine Readme.txt Datei, in der ausführlich beschrieben wird, wie die Software zu starten ist! Zudem ist der aktuelle Dokumentationskopf abzugeben. Die Dateien sind als *.zip Ordner (mit cc an den/die Teampratner_in) per E-Mail abzugeben. Die Abgabe gehört zu den PVL-Bedingungen und ist einzuhalten, terminlich wie auch inhaltlich!

Wird eine Aufgabe nicht erfolgreich bearbeitet, gilt die **PVL** als **nicht bestanden**. Damit eine Aufgabe als erfolgreich gewertet wird, muß die Befragung als erfolgreich gewertet werden sowie die Abgabe abgenommen worden sein. **Alle gesetzten Termine sind einzuhalten**. Dies ist notwendig, da sonst erhebliche zeitliche Verzögerungen stattfinden würden..