MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG242 Programming Language Concepts – Spring 2021 - Lab Submission 2

Date handed out: 3 May 2021, Monday 18:00
Submission due: 4 May 2021, Tuesday 23:55

Please Read This Page Carefully

Submission Rules

1. **You need to write a comment on the first line of your file**, stating that you read the rules specified here and the submission is your own work. **Submissions without this statement will not be graded.**
   Example, "-- I read and accept the submission rules and this is my own work that is done by myself only"
2. As explained in syllabus[1] provided for CNG 242, attempting any academic dishonesty[2], breaking professionalism and ethics[3] rules may result in the following:
   - You might be asked to perform an oral test to confirm your submission.
   - You may receive a "zero" grade for this submission
   - **You may receive "zero" from all future submissions.**
   - You may receive a failing letter grade for the course and this case might be forwarded to the discipline committee.
3. You cannot use someone else's code.
4. You **cannot hire** someone to write the code for you.
5. **You cannot share your code with someone else.**
6. You need to be ready to demonstrate your own work, answer related questions, and have short coding sessions if necessary.
7. You need to submit a **single Haskell file named with your student id** only. For example, **1234567.hs**
8. Function names must be the same as the provided questions.
9. You cannot share this worksheet with any third parties. Upon doing so, any detected action will directly be sent to the disciplinary committee.
10. **You CAN import modules, any module you import will be accepted. You can also use any function there are no limits. However, keep in mind that, you should be able to demonstrate everything you used in a demo session.**
11. You should only submit one solution per question. Your solution might have multiple lines. **Only the functions with the same name will be graded.**
12. You can only get full marks, if your file fully compiles, runs, and generates correct output for all possible cases. **Non-general static solutions will not be graded!**
13. **If you are added to another section as a guest student**, you will still have to submit your submission under the main section you are registered. You can check your registered section by trying to see your grades in ODTUClass. Only submit your file to the section that you can see your grades.

[1] Page 3&4 (Course rules, #1,2,3)
[2] Taking unfair advantage in assessment is considered a serious offence by the university, which will take action against any student who contravenes the regulation through negligence or deliberate intent.
[3] For a comprehensive cheating definition, please refer to: https://ncc.metu.edu.tr/res/academic-code-of-ethics . When a breach of the code of ethics occurs (cheating, plagiarism, deception, etc.), the student will be added to the BLACKLIST.

Questions:

Rules:

- Strictly obey the specifications, input output formats. Do not print extra things.
- **You have to include at least one function from a module. Therefore, you need to import at least one module for your submission to be graded!**

1. **[0.4 Marks]** Implement a function called **lastncompare** which takes two strings and a number n, and then compares the last n elements of the lists. This comparison is not case-sensitive.

   Sample Run:
   lastncompare "CNG242" "cng140" 3
   False

   lastncompare "CNG242" "Helloeveryonehopeyouareok242" 3
   True

2. **[0.6 Marks]** Write a Haskell function **capitalizeFirstLast** that takes a sentence, makes the first and last letters of each word upper case and makes the rest of the letters lower case.

   Sample Run:
   capitalizeFirstLast "Hello world"
   "HellO WorlD"

   Sample Run:
   capitalizeFirstLast "242 Programming Language Concepts"
   "242 Programming Language ConceptS"

3. **[0.6 Marks]** Write a Haskell function **verifier** that takes a password and checks whether or not it is valid. The password should contain at least 7 at most 10 characters, 2 lower case letters, 2 upper case letters and a number.

   Sample Run:
   verifier "ABcd13ddcA"
   True

   verifier "ABcd13ddcAd"
   False

4. **[0.4 Marks]** Modify your file to create a module named exactly **Qfour** from these three functions you implemented above. Upon calling the module from your file, only the lastncompare, capitalizeFirstLast and verifier should be visible. Your helper functions should not be visible. **Tip:** if you want to import a module to use inside your module, you should define your module first and then you should import the pre-implemented modules.

Assessment Criteria

The assignment will be marked as follows:

Item	Marks (Total 2)
Q1,2,3,4 – Trying something relevant	0.1
Q1 – Sample run cases works	0.2
Q1 – Works for all appendix + Hidden inputs	0.4
Q2,3 – At least one of the sample runs work	0.2
Q2,3 – Both sample runs and some appendix examples work	0.3
Q2,3 – Works for all appendix but not for all hidden inputs	0.5
Q2,3 – Fully works	0.6
Q4 – Fully meets the criteria otherwise no partial points.	0.4

Appendix

```
*Qfour> {- Q1 EXAMPLES BEGIN -}
*Qfour> lastncompare "CNG242" "cng140" 2
False
*Qfour> lastncompare "CNG242" "cng140" 1
False
*Qfour> lastncompare "CNG242" "cng140" 0
True
*Qfour> lastncompare "book" "cook" 5
False
*Qfour> lastncompare "book" "cook" 4
False
*Qfour> lastncompare "book" "cook" 1
True
*Qfour> lastncompare "book" "cook" 0
True
*Qfour> lastncompare "bOoK" "CoOk" 4
False
*Qfour> lastncompare "bOoK" "CoOk" 1
True
*Qfour> lastncompare "COOK" "cook" 5
True
*Qfour> {- Q1 EXAMPLES END -}
*Qfour>
*Qfour> {- Q2 EXAMPLES BEGIN -}
*Qfour> capitalizeFirstLast "Bohemian rhapsody"
"BohemiaN RhapsodY"
*Qfour> capitalizeFirstLast "Nothing ElsE matters"
"NothinG ElsE MatterS"
*Qfour> capitalizeFirstLast "Another Brick in thE wall 1979"
"AnotheR BricK IN ThE WalL 1979"
*Qfour> capitalizeFirstLast "Ode to joy"
"OdE TO JoY"
*Qfour> capitalizeFirstLast "242"
"242"
*Qfour> capitalizeFirstLast "a little night mUSIc"
"A LittlE NighT MUSIC"
*Qfour> capitalizeFirstLast "Sweet Child o' Mine"
"SweeT ChilD O' MinE"
*Qfour> capitalizeFirstLast "a very long SENTENCE which ends with A dot."
"A VerY LonG SENTENCE WhicH EndS WitH A Dot."
*Qfour> {- Q2 EXAMPLES END -}
*Qfour>
```

```
*Qfour>
*Qfour> {- Q3 EXAMPLES BEGIN -}
*Qfour> verifier "1234567"
False
*Qfour> verifier "abcdefg"
False
*Qfour> verifier "ABCDEFG"
False
*Qfour> verifier "ABcd_-7"
True
*Qfour> verifier "ABc__67"
False
*Qfour> verifier "Cng242-ProgrammingLanguageConcepts"
False
*Qfour> verifier "Cng242-PLC"
True
*Qfour> verifier "Cng242-P.L.C."
False
*Qfour> verifier "OdeToJoy13"
True
*Qfour> {- Q3 EXAMPLES END -}
*Qfour>
*Qfour> {- Q4 EXAMPLES BEGIN -}
*Qfour> --assume you double clicked the file to open,
*Qfour> helper "This is just a blank function, you don't have to implement this one"
"This is a demo helper function hidden from the module"
*Qfour> --loading the module now (In the line below)
*Qfour> :module Qfour
Prelude Qfour> helper "Since it doesn't belong to my module it won't be visible anymore"

<interactive>:42:1: error:
    Variable not in scope: helper :: [Char] -> t
Prelude Qfour> {- Q4 EXAMPLES END -}
```

```
helper x = "This is a demo helper function hidden from the module"
```

APPENDIX EXAMPLES IN RAW TEXT

{- Q1 EXAMPLES BEGIN -}
lastncompare "CNG242" "cng140" 2
lastncompare "CNG242" "cng140" 1
lastncompare "CNG242" "cng140" 0
lastncompare "book" "cook" 5
lastncompare "book" "cook" 4
lastncompare "book" "cook" 1
lastncompare "book" "cook" 0
lastncompare "bOoK" "CoOk" 4
lastncompare "bOoK" "CoOk" 1
lastncompare "COOK" "cook" 5
{- Q1 EXAMPLES END -}

{- Q2 EXAMPLES BEGIN -}
capitalizeFirstLast "Bohemian rhapsody"
capitalizeFirstLast "Nothing ElsE matters"
capitalizeFirstLast "Another Brick in thE walL 1979"
capitalizeFirstLast "Ode to joy"
capitalizeFirstLast "242"
capitalizeFirstLast "a little night mUSIc"
capitalizeFirstLast "Sweet Child o' Mine"
capitalizeFirstLast "a very long SENTENCE which ends with A dot."
{- Q2 EXAMPLES END -}

```
{- Q3 EXAMPLES BEGIN -}
verifier "1234567"
verifier "abcdefg"
verifier "ABCDEFG"
verifier "ABcd_-7"
verifier "ABc__67"
verifier "Cng242-ProgrammingLanguageConcepts"
verifier "Cng242-PLC"
verifier "Cng242-P.L.C."
verifier "OdeToJoy13"
{- Q3 EXAMPLES END -}

{- Q4 EXAMPLES BEGIN -}
--assume you double clicked the file to open,
helper "This is just a blank function, you don't have to implement this one"
--loading the module now (In the line below)
:module Qfour
helper "Since it doesn't belong to my module it won't be visible anymore"
{- Q4 EXAMPLES END -}
```