Prof. Dr. Visvanathan Ramesh
Alperen Kantarcı
Gamze Akyol

Goethe-Universität Frankfurt am Main
Institut für Informatik

Programmierparadigmen und Compilerbau

Sommersemester 2023

Exercise Nr. 6

Deadline Wednesday 19.07.2023, 10:00 in Moodle!

Question 0 **(0 Points)**

Read the following tips!

- Group work is **not allowed!** You can talk and discuss with your friends about the exercises, but you should solve the exercises **individually**. Plagiarism is not tolerated.
- You can use ChatGPT or other web sources but you cannot ask the whole question. You should be able to explain every step of your code.
- Upload all the files you want to submit into **a single .zip file on Moodle**. Please ensure that you **only submit .pdf and .hs files.**
- Please write your **matriculation number**, **name** and **group number** to each file that you are submitting.
- Your submission file name should be in the following order:

  Name_matriculationNumber_Group_GroupNumber_Exercise_ExerciseNumber.extension.

  For example **AlperenKantarci_1111111_Group_6_Exercise_6.zip**

Question 1 - Shift Reduce Parser **(2 + 3 + 2 + 3 = 10 Points)**

Consider the following grammar where [S,C] are Non-terminals and [x,y] are terminals.

```
S → CC
C → Cx | y
```

Your task is to manually parse the string 'yxxy' using a bottom-up, shift-reduce parsing approach. Assume that the parser operates by shifting input symbols from left to right onto a stack and reducing them to higher-level grammatical structures according to the provided grammar rules.

1. Explain the initial state of the parser (i.e., the state of the stack and the remaining input before parsing begins).
2. Perform the sequence of shift and reduce operations required to parse the string 'yxxy'. For each operation, explain what the parser does and why, and describe the state of the stack and the remaining input after the operation.
3. If the string 'yxxy' can be successfully parsed according to the provided grammar, explain the final state of the parser (i.e., the state of the stack and the remaining input after parsing completes). If the string cannot be parsed, explain why not.
4. Construct the corresponding parse tree for the string 'yxxy' based on the operations you performed in step 2. Each leaf of the tree should correspond to a terminal symbol in the input string, and each internal node should correspond to a non-terminal symbol in the grammar.

Question 2 - Stack **(5 Points)**

Consider a pure stack machine and instruction $r = F(a_1, a_2, ..., a_n)$. It gets n operands from the stack, computes the operation F with the operands and pushes the result onto the stack.

Illustrate the steps of a pure stack machine to calculate the following expression:

$$2 * 6 + 8\%4 - 24/8$$

Assume that the set of possible F's includes binary operators '`add`', '`multiply`', '`substract`', '`divide`', '`modulo`' in addition to the `pop`, `push i` (i.e. `push 0` is push the first element of stack to the stack), and `pushK` (push constant).

Question 3 - Register Reorganisation **(5 Points)**

Consider the following register operations:

```
[LoadOp "%r1" (Const 12)
     ,LoadOp "%r2" (Const 5)
     ,LoadOp "%r3" (Const 7)
     ,LoadOp "%r4" (Const 2)
     ,PrimOp "*" "%r5" ["%r1", "%r2"]
     ,PrimOp "/" "%r6" ["%r3", "%r4"]
     ,PrimOp "+" "%r7" ["%r5", "%r6"]]
```

What is the minimum number of registers required to execute these commands? Use the lifetime analysis of variables to show the solution. Can you minimize the number of registers used by rearranging the sequence of commands without changing the final result? If possible, write the new sequence of operations that uses minimum number of registers.