**CSCI 465 Fall 2020**

Exam 2: **TAKE-HOME**

Total Points: 120

Total number of questions: 7

**Due date: December 15 at 12:00 p.m.**

Name:\_\_\_Derek Trom\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Note: Your answers must be typed and spellchecked. To answer the questions, you are allowed to consult textbook, lecture notes, your own note, published articles in Journals/Conferences. However, this is individual work meaning that you are NOT allowed to share your solution with other students whatsoever. Also, you are encouraged to use proper citations/references to refer the resources that you have read/used to formulate your solutions.*

1. ( 10 points)You need to briefly answer the following questions regarding your the compiler project:
   1. What was the main obstacle when you first started working on your project?
   2. Which delivery was the easiest one and how long did it take you to do it?
   3. Which delivery was the hardest part and how long did it take you to do it?
   4. Of which part of your project is the most rewarding part?
   5. Which part of your compiler would you like to re-implement if you had the chance to improve *your compiler*?
   6. What are the lessens you learn from your compiler? What would you do differently if you are asked to construct yet another compiler next time?
   7. What type of discussion would like to be added/deleted from the course so the course becomes more interesting and meaningful?
   8. How big is your compiler in terms of line of code (LOC)?
   9. If you are given enough time, what would be the additional features you would like to add to your compiler?
   10. What is the most important property of a compiler? How did you validate your compiler according to this property?
2. (15 points) For each of the following grammar indicate whether overall, general attribute value flow is bottom-up, top-down, left-to-right, and right-to-left.

**(a).**

G→A↓l

A↓n→B↓3n A↓7n

→”c” C↓n-1

B↓n→”a” B↓n+4 “b” C↓2n

→”b”

→”c”

**(b)**

G→A↑x

A↑n→B↑u ↑v A↑y [x=uy+v]

→”c” C↑z [x=2z]

B↑v→”a” B↑r↑s“b” C↑x [u=2r+x-s; v=s+1]

→”b” [u=1; v=2]

C↑x →”c” [x=3]

**(c)**

G →A↓ 0 ↑r

A↓x↑z→B↓y ↑z A↓ x↑y

→”c” C↓x↑y [z=10y+3]

B x w → “a” B↓10y+2↑z “b” C↓x↑y [w=10z+1]

→”b” [w=10x+2]

C x y →”c” [y=10x+3]

3. (20 points) consider the following grammar:

Terminals = {a, b,c}

Non-Terminals = {S, A}

**Rules:**

(1) S → AA

(2) S→ bc

(3) A→ baA

(4) A→ c

Suppose also we have the following LR(0) states and transitions for the above grammar:



1. Create the LR(0) parse table
2. Is the above grammar LR(0)? (explain briefly)
3. (10 points) The following declarations are given for a language that uses name equivalence:

A, D: array [1..100] of int;

C: array [1..100] of int;

F: array [1..100] of int;

Explain briefly: 1) which of the above four variables have the same type, and 2) which ones have different types?

1. (15 points) Intermediate Representations (IRs) plays a significant role in translating diffident languages working on different platforms.
   1. What are the main properties of Intermediate Representations (IRs)?
   2. Moreover, standardizing IRs has been suggested to address two constant issues in computing industry: 1) software compatibility, and 2) compiler interoperability. Briefly explain how the standardized IRs will address those issues.

**Note**: the complete discussion of the proposed standardization of IR can be found in an article published by Communication of ACM, December 2013, Vol.6, N0.12. The article is available on the blackboard system for the course and can be found under “resources”.

6. (20 points) strongly typed, statically checked languages can help the programmer produce valid programs by detecting large classes of erroneous programs.

a) In what way this feature can improve the compiler’s ability to generate efficient code for a program?

b) Some programming languages either omit declarations or treat them as optional Information. Examples include Scheme program that lacks declarations for variables. Therefore, in the absence of declarations, what element of type system can be used to determine a type for each variable, and in what way it makes harder to implement the language (briefly explain).

7. ( 30 points) Use the following translation scheme (SDT) and the type **float** [3][4] (i.e., two-dimensional array of type **float**) to perform the following tasks:

1. Generate the parse tree
2. Annotate the tree with attributes (i.e., type and width)
3. Show how the width of the type **float** [3][4] is computed

**Productions Syntax Directed Translation**

**-----------------------------------------------------------**

T🡪 B {t=B.type; w=B.width;}

C {T.type = C .type; T.width = C .width;}

B🡪 **int** {B.type = integer; B.width = 4;}

B🡪 **float** {B.type = float; B.width = 8;}

C🡪 ε {C.type = t; C.width = w;}

C🡪 [**num**] C1 {C.type = array(**num**.value, C1.type); C.width=**num**.value × C1.width; }