BRAC UNIVERSITY
Department of Computer Science and Engineering

Examination: Final
Duration: 1 Hour 20 Minutes

Set - B

Semester: Fall 2024
Full Marks: 40

CSE 420: Compiler Design

Figures in the right margin indicate marks.

Answer all the questions

<table>
<tr><th>COs</th><th>Questions</th><th>Marks</th></tr>
<tr><td>C05</td><td>1. Consider the following SDD. Draw the Annotated Parse Tree and Evaluate it for the String: ((a > b) && (c > d)) || ((w < x) && (y < z))

Note: Here newLabel() method creates a label which is ‘L:’ with a subscript 1, 2, 3 and so on. For example, the first time the method is called it will create L_1:, next time it is called it will create L_2: and so on.

<table>
<tr><th>PRODUCTION</th><th>SEMANTIC RULES</th></tr>
<tr><td>P → (B)</td><td>B.true = newlabel()
B.false = newlabel()
P.code = B.code</td></tr>
<tr><td>$B \rightarrow B_1 \;||\; B_2$</td><td>$B_1.true = B.true$
$B_1.false = newlabel()$
$B_2.true = B.true$
$B_2.false = B.false$
$B.code = B_1.code \;||\; label(B_1.false) \;||\; B_2.code$</td></tr>
<tr><td>$B \rightarrow B_1 \;\&\&\; B_2$</td><td>$B_1.true = newlabel()$
$B_1.false = B.false$
$B_2.true = B.true$
$B_2.false = B.false$
$B.code = B_1.code \;||\; label(B_1.true) \;||\; B_2.code$</td></tr>
<tr><td>$B \rightarrow (\, B_1)$</td><td>$B_1.true = B.true$
$B_1.false = B.false$
$B.code = B_1.code$</td></tr>
<tr><td>$B \rightarrow E_1$ rel E_2</td><td>$B.code = E_1.code \;||\; E_2.code$
$\;\;\;\;||\; gen('\texttt{if}'\; E_1.addr\; \textbf{rel}.op\; E_2.addr\; '\texttt{goto}'\; B.true)$
$\;\;\;\;||\; gen('\texttt{goto}'\; B.false)$</td></tr>
<tr><td>$E \rightarrow$ id</td><td>$E.addr = top.get(\textbf{id}.lexeme)$
$E.code = ''$</td></tr>
</table>
</td><td>10</td></tr>
</table>

CO4	2. Consider the following *SDT*. Draw the **Annotated Parse Tree** and **Evaluate** the values of the attributes **type** and **width** along with variable **offset** next to the appropriate non-terminal nodes for the following String: **float [4] [5] x; record {int a; float b;} y; int z;** Afterwards draw the memory diagram along with actual location and data **Note: Offsets are given in decimal. Memory location is also given in decimal. Every memory unit is of 1 byte. The width of int and float is also given in byte in the SDT. The variable memory_addr refers to the starting position of the memory address where the variables will be stored.** SDT:	15

$P \rightarrow \{\ offset = 0;$ memory_addr = 58342;$\}\ D$

$D \rightarrow T\ \mathbf{id}\ ;\ \{\ top.put(\mathbf{id}.lexeme,\ T.type,\ offset);\ \ offset\ =\ offset + T.width;\ \}\ D_1$

$D \rightarrow \epsilon$

$T \rightarrow \mathbf{record}\ '\{'\ \{\ Env.push(top);\ top = \mathbf{new}\ Env();\ \ Stack.push(offset);\ offset = 0;\ \}\ \ D\ '\}'$ $\{\ T.type = record(top);\ T.width = offset;\ top = Env.pop();\ offset = Stack.pop();$

$T \rightarrow B\{\ t = B.type;\ w = B.width;\ \}C$ {T.type = C.type; T.width = C.width;}

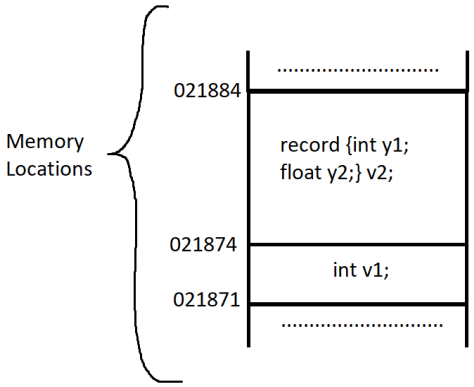
$B \rightarrow \mathbf{int}$ $\{\ B.type = integer;\ B.width = 3;\ \}$

$B \rightarrow \mathbf{float}$ $\{\ B.type = float;\ B.width = 7;\ \}$

$C \rightarrow \epsilon$ $\{\ C.type = t;\ C.width = w;\ \}$

$C \rightarrow [\ \mathbf{num}\ ]\ C_1$ $\{\ C.type = array(\mathbf{num}.value,\ C_1.type);\ C.width = \mathbf{num}.value \times C_1.width;\ \}$

Example of Memory Diagram for String: int v1; record {int y1; float y2; } v2; where memory_addr = 021871; in the SDT.



C05	3. Consider the following **SDD.** Draw the **Annotated Parse Tree** and **Evaluate** it for the String: **if (a > b) a = c + d ; else a = a + b ;** **Note: Here new Temp() creates an instance of compiler generated temporary variables which is 't' with a subscript 1, 2, 3 and so on. For example, the first time an instance is created it will create t_1, next time it is called it will create t_2 and so on. Furthermore, the newLabel() method creates a label which is 'L:' with a subscript that follows similar logic as the temporary variable, for example, L_1:, L_2: and so on.**	10
C04	4. Explain the workings of SDT. Furthermore, explain why we use SDT. You are free to use a small example to clarify your explanation if necessary.	5

PRODUCTION	SEMANTIC RULES
$P \rightarrow S$	$S.next = newlabel()$ $P.code = S.code \mid\mid label(S.next)$
$S \rightarrow \textbf{id} = E\ ;$	$S.code = E.code \mid\mid$ $gen(top.get(\textbf{id}.lexeme)\ '='\ E.addr)$
$S \rightarrow \textbf{if}\ (\ B\ )\ S_1\ \textbf{else}\ S_2$	$B.true = newlabel()$ $B.false = newlabel()$ $S_1.next = S_2.next = S.next$ $S.code = B.code$ $\mid\mid label(B.true) \mid\mid S_1.code$ $\mid\mid gen('\texttt{goto}'\ S.next)$ $\mid\mid label(B.false) \mid\mid S_2.code$
$B \rightarrow E_1\ \textbf{rel}\ E_2$	$B.code = E_1.code \mid\mid E_2.code$ $\mid\mid gen('\texttt{if}'\ E_1.addr\ \textbf{rel}.op\ E_2.addr\ '\texttt{goto}'\ B.true)$ $\mid\mid gen('\texttt{goto}'\ B.false)$
$E \rightarrow E_1 + E_2$	$E.addr = \textbf{new}\ Temp\,()$ $E.code = E_1.code \mid\mid E_2.code \mid\mid$ $gen(E.addr\ '='\ E_1.addr\ '+'\ E_2.addr)$
$\mid\ \textbf{id}$	$E.addr = top.get(\textbf{id}.lexeme)$ $E.code = ''$