CS 403/503, Programming Languages
Spring 2021
Exam One

Date: 02/22/2021

Name: Eland Anthony

1). Basic Concepts

a). Name two solutions to the dangling pointer problem. **(4 points)**

Utilizing tombstones to safely deallocates pointers by being assigned invalid values and prevents a pointer from ever pointing to a deallocated variable.

Lock and Key Method, which uses lock values and the dispose to maintain its address value but clear the lock value.

b). Name two methods to reclaim garbage. **(4 points)**

The eager approach to reclaiming garbage utilizes reference counters. Reference counters maintain a counter that stores the number of pointers that are currently pointing at the cell for each cell.

The lazy approach to reclaiming garbage, known as the mark-sweep operation.

c). Name the three things added to a context-free grammar to make an attribute grammar. **(6 points)**

- attributes,
- attribute computation functions,
- and predicate functions.

d). Given the following C program, what is the value of z after the assignment statement in main? **(6 points)**

- assuming operands are evaluated from left to right.
  10
- assuming operands are evaluated from right to left.
  20

```
int x = 5;

int fun(int i) {
        int y = x;
        x = x*i;
        return y;
}
int main() {
        int z = x + fun(2);
        . . .
}
```

2). (20 points) Consider the following program, written with the syntax of JavaScript.

```
function sub() {
   var x;
   function sub1() {
         var x;
         function sub2() {
               var x;
               x = 20;
               sub3();
         }
         x = 10;
         sub2();
   }
   function sub3() {
         document.write("x = " + x + "");
   }
   x = 5;
   sub1();
}
```

- Under static-scoping, what value of x is displayed in function `sub3`?
  5
- Under dynamic-scoping, what value of x is displayed in function `sub3`?

20

3). (20 points) Eliminate the `goto` statements in the following pseudocode segment and rewrite it using a loop structure in C, C++, Java, or C#.

```
        n = 191
        k = 2
loop:
        if k >= n then goto out
        r = n % k
        if r is 0 then goto out
        k = k + 1
```

```
         goto loop
out:
         . . .
```

Answer:

C++

```
int n = 191;
int k = 2;
int r = 0;

while (k < n) {
   r = n % k;
   if (r != 0)
      k = k + 1;
}
```

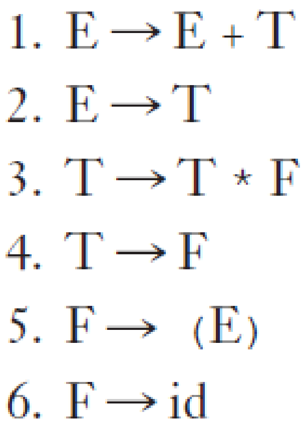
4). (20 points) The following grammar is ambiguous. Rewrite it to make it unambiguous.

```
<A> → <A> .and. <A> | (<B>)
<B> → <B> .or. <B> | <C>
<C> → T | F
```

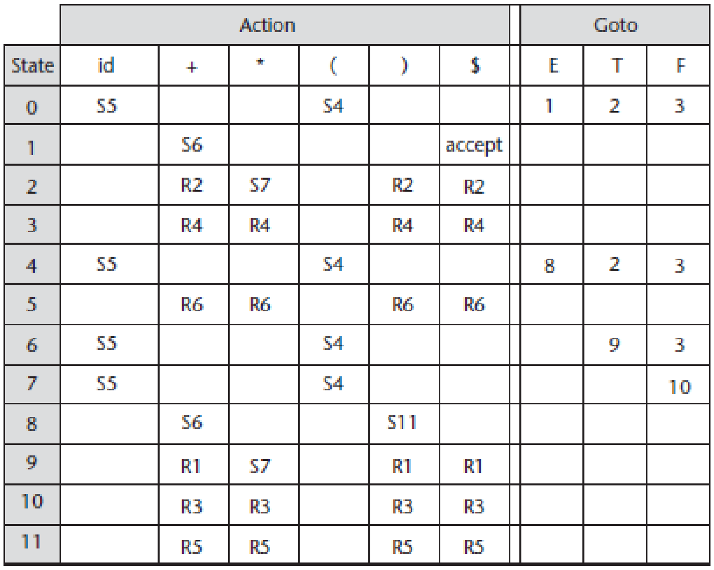
Answer:

```
A -> A * T
T -> A | B
B -> B | F
F -> B | C
C -> T | F
```

5). (20 points) Given the following grammar and the corresponding LR parsing table, complete the next eight (8) LR parsing actions.



1. E → E + T
2. E → T
3. T → T * F
4. T → F
5. F → (E)
6. F → id



	Action						Goto		
State	id	+	*	(	)	$	E	T	F
0	S5			S4			1	2	3
1		S6				accept			
2		R2	S7		R2	R2			
3		R4	R4		R4	R4			
4	S5			S4			8	2	3
5		R6	R6		R6	R6			
6	S5			S4				9	3
7	S5			S4					10
8		S6			S11				
9		R1	S7		R1	R1			
10		R3	R3		R3	R3			
11		R5	R5		R5	R5			

Stack	**Input**	Action
0T2*7(4E8	+ id) $	Shift 6
0T2*7(4E8+6	Id) $	Shift 5
0T2*7(4E8+6id5	) $	Reduce 6[6, F]
0T2*7(4E8+6F3	) $	Reduce 4[6, T]
0T2*7(4E8+6T9	) $	Reduce 1[4, E]
0T2*7(4E8	) $	Shift 11
0T2*7(4E8)11	$	Reduce 5[7, F]
0T2*7F10	$	Reduce 3 [0, T]
0T2	$	Reduce 2 [0, E]
0E1	$	accept