Assignment 4 part2
Junhan Liu 7228243
YING JIN GUAN 7285769
CSI 3120

1. Consider a two-dimensional array with 7 columns and 8 rows stored starting at location 2001 in row-major order. Its elements occupy 4 bytes each. The location of the 3rd element of the 4th row is:

Assume the starting index of the array is A[1,1], and location of A[1,1] = 2001.
We calculate A[4,3].
A[4,3] = 7*(4-1)+(3-1) = 23
The location of A[4,3] = 23*4+2001 = 2093

2,
Type checking is the process of verifying and enforcing the constraints of types, and it can occur either at compile time (i.e. statically) or at runtime (i.e. dynamically).
A language is statically-typed if the type of a variable is known at compile time instead of at runtime.
Dynamic type checking is the process of verifying the type safety of a program at runtime.
The big benefit of static type checking is that it allows many type errors to be caught early in the development cycle.
In contrast to static type checking, dynamic type checking may cause a program to fail at runtime due to type errors.
Dynamic type checking typically results in less optimized code than does static type checking; it also includes the possibility of runtime type errors and forces runtime checks to occur for every execution of the program (instead of just at compile-time). However, it opens up the doors for more powerful language features and makes certain other development practices significantly easier.

3. What are the advantages and disadvantages of the ability to change objects in Ruby (language evaluation criteria)?
Advantage: object is easy to pass since it can be given any size of lense, for example, array-type in ruby is dynamic with length therefore, no restriction for change array. Ruby program can change the class definition while it is running.
Disadvantage: Ruby is slow programming language so process of changing object will cost more time.

4,
```
common([],_,[]):- !.

common([H|T], L, R):-
        (member(H, L)
        -> R = [H|RR],
```

common(T, L, RR)
;common(T, L, R)).

5. Using labels and gotos give an operational semantic definition of the C for, while structures.

C Statement :

for (expr1; expr2; expr3) {
. . .
}

Meaning:

expr1;
loop: if expr2 == 0 goto out
. . .
expr3;
goto loop
out: . . .

C Statement :

while (expr1) {
expr2
}

Meaning:

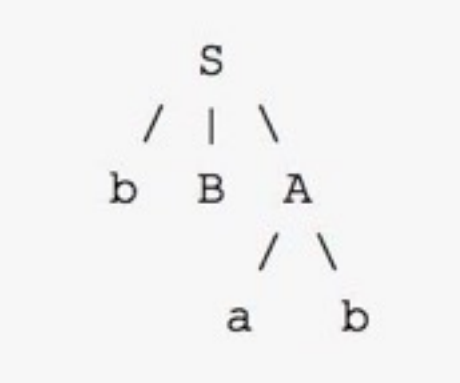
expr1;
loop: if (expr1) goto expr2
loop2: if (!expr1) goto out
out: . . .

6, a) aaAbb

phrase: aaAbb, aAb, b

simple phrase: b
handle: b

b)



phrase: bBab, ab
simple phrase: ab
handle: ab

c) aaAbBb

aaAbBb --> aSBb --> aSBB --> x
or aaAbBb--> aaAbBB --> aSBB --> x
The string cannot be drived by the grammar.

7,
“ø” means empty.
a)
E → TE’
E’ → +TE’ | !E’ | ø
T → int | (E)

b)
L → XL’
L’ → ;XL’ | ø
X → int | string | (L)

c)
P → pP’
P’ → H4UP’ | ø
H → h
U → u | uP

8,
8-bit string with central pari is “11” can be derived from any four RHS of S,
i.e. 0S0, 1S0, 0S1, 1S1
Each one of the four can choose any one of the four grammar to generate the final string.
Thus, each one can generate 16 strings, and there are four of them.
So a total 64 strings can be generated.

9.
T->ε |T|(T) |IOI|IOT|TOI
I->I|a|b|c
O->| ⊙| ⊗ |⊕

10,

a) a > b XOR c OR d <= 17
=> (((a>b)1 XOR c)3 OR (d <= 17)2)4

b) -a + b
=> (- (a + b)1)2

11.
a. a * b - 1 + c
=>((a * b) - 1) + c)

b. a * (b - 1) / c **mod** d
=>(((a*(b-1))/c)mod d)

c. (a - b) / c & (d * e / a - 3)
=>(((a-b)/c) &(((d*e)/a)-3))

d. -a **or** c = d **and** e
=>((-a)or((c=d)ande))

12,
a) left -> right
sum1 = (10/2) + fun(10)
= 5 + { k = 14; return 3*14-1;}
= 46

sum2 = fun(10) + (14/2)
=>41 + 7
=> 48

b) right -> left
sum1 = (i/2) + fun(10)
= (i/2) + { i = 14; return 41}
= 7 + 41
= 48

sum2 = fun(10) + (10/2)
= 41 + 5
=46

13,
static: X is 5
dynamic: X is 10

14,
Sub1: a(sub1), y(sub1), z(sub1), x(main)
Sub2: a(sub2), b(sub2), z(sub2), y(sub1), x(main)
Sub3: a(sub3), x(sub3), w(sub3), y(main), z(main)

15.Show the stack with all activation record instances, including static and dynamic chains, when execution reaches position 1 in the following skeletal program. Assume Bigsub is at level 1.
STACK:

<table>
<tr><td>local variable</td><td rowspan="6">D</td></tr>
<tr><td>parameter</td></tr>
<tr><td>static link</td></tr>
<tr><td>dynamic link</td></tr>
<tr><td>return to A</td></tr>
<tr><td>local variable</td><td></td></tr>
</table>

parameter	C
static link	
dynamic link	
return to A	
local variable	A
parameter (flag=false)	
static link	
dynamic link	
return to A	
local variable	B
parameter	
static link	
dynamic link	
return to A	
local variable	A
parameter (flage=true)	
static link	
dynamic link	
return address	
local variable	Bigsub
parameter	

static link
dynamic link
return

16,
Busy-waiting or spinning is a technique in which a process repeatedly checks to see if a condition is true, such as whether keyboard input or a lock is available. Spinning can also be used to generate an arbitrary time delay, a technique that was necessary on systems that lacked a method of waiting a specific length of time. Processor speeds vary greatly from computer to computer, especially as some processors are designed to dynamically adjust speed based on external factors, such as the load on the operating system. Busy waiting may loop forever and it may cause a computer freezing.
The main problem would be the CPU resources that area allocated to the waiting for the task, if it were to be suspended, the CPU can then be used for other purposes.

17.
Suppose two tasks, A and B, must use the shared variable Buf_Size. Task A adds 2 to Buf_Size, and task B subtracts 1 from it. Assume that such arithmetic operations are done by the three-step process of fetching the current value, performing the arithmetic, and putting the new value back. In the absence of competition synchronization, what sequences of events are possible and what values result from these operations? Assume that the initial value of Buf_Size is 16.

Event:
A : fetch current value of Buf_Size -> Buf_Size=16->Perform operator 16+2=18-> put 8 to Buf_Size;
B : fetch current value of Buf_Size -> Buf_Size=18->Perform operator 18-1=17-> put 17 to Buf_Size;
the value result is 18 from these operation.