1. Write a BNF grammar for the **for** loop statement of **C**. ~~Assume this is the 89 standard that does NOT allow new variables as C++ did~~ (not allowed is the **int** in **for (int i=0; i<5; i++)**. Also assume block is NOT a statement (different books take different take on this). Further assume <expression> and <statement> and <block> nonterminals are already defined (you dont' define them, just use them). Do not use the extended BNF notation.

Answer:

<for loop>::=(<expression>; <expression>; <expression>)<block>

1. For the above, write informal operational semantics (as you would give to someone new to C, trying to explain what it is)

Answer: B := true | false | B & B | B or B | not B | exp < exp | exp = exp

E := 0 | 1 | 2… | N | I | E + E | E \* E | E - E | -E  
for (1 := N); (if B); (2 := E) do (statement)

B is Boolean expression and E is arithmetic expression. Check if B is true, and if so do <statement>, else skip (statement). After (statement) perfom another loop if B is still true after (1 := N) is assigned a new value (2 := E) after performing the first iteration.

1. Design a regular language where every sentence has to start with any number of strings 101 (any number is none or more), then repeats 00 any number of times, then repeats 01 at least once.

Answer: (101) \* (00) \* (01) \*

1. For the above, how many valid strings there are?

Answer: Infinite strings are valid for above. Could always add on

1. For the above, how many valid strings are there if we restrict the total length to a) at most 7 bits? b) exactly 7 bits?

Answer: a) 27 valid strings b) 27 valid strings

1. If you were designing a new programming language to bring to market, would you make it regular, context free, context sensitive, or unrestricted and why?

Answer: every regular grammar is context free grammar. Regular and context free grammars are recursive. Context free are the most that can be effectively processed. I would not use context sensitive grammar because of the lack of parsing algorithms. I would design a context free language to bring to market.

1. If translation of your programs was not your concern at all (aliens offered to do it for you), would the above answer change, and how and why?

Answer: An interpreter steps through the source code line by line, figuring out what it’s doing as it goes. If translation was not a problem, I would use unrestricted grammar. I would do this because all strings and combinations of strings are allowed in unrestricted grammar. Also, unrestricted grammars include all other types of grammar, so I would not lose any functionality.