1.1:

(A): int 18 = 18;

(B): int x == 4;

(C): char[] = new char['h'];

(D): public void repeat(int x) {

repeat(7);

}

(E): while (flag = true) {

// this will run but the boolean check is incorrect, should be '=='

}

1.8:

This dependence management is correct because if files necessary for a program to run are altered, the program will need to recompile. An unnecessary recompile would happen when comments are changed in a source file; this doesn't change the functionality of the code but the makefile would still need to recompile. The files won't recompile when there is an existing error in any of the source files.

2.1:

(A): C example strings: "Hello World", "Joe", "34"

x → (a-z)(A-Z)(0-9)

strings → "(x + \" + \\)" *

(B): Pascal example comments: (* Hello World *), { another example }

x → (a-z)(A-Z)(0-9)

comments → (* ^/(?!{} | ?!()) x *) | { ^/(?!{} | ?!()) x }

(C): C numeric constant examples: 123, 026, 0xb3

zero → 0

decimal → (0-9)

hex → 0x

octal → (0-7)

rexpr → (hex + octal) * | decimal | (zero + octal*)*

2.13:

(A) :

```
                                   stmt
                                     /\
                   assignment              subr_call
                   /                                /\
          id (a)                           id       ( arg_list )
                                           /                 /\
                                        foo             a          b
```

(B) :

stmt → assignment

→ id(a)

→ subr_call

→ id

→ foo

→ arg_list

→ a

→ b

2.17:

while → “while” “(“ stmt “)” “{“ stmt “}”

if → “if” “(“ stmt “)” “{“ stmt “}”