Lab 7

4.1

Token *cur (in tokenize)
char *p (in tokenize)
return Pointer
frame Pointer
char *p
char buf[32];
int len
Token *cur (points to cur in tokenize)
char *start (points to p in tokenize)

In this case the local variables will be saved below the frame pointer and the return pointer in the stack. The buffer is vulnerable to an buffer overflow since it can rewrite these important control elements.

4.2

Valgrinds finds the buffer overflow vulnerability and stops the program. There is no limit to how much can be written into the buffer in this function which is what causes the problem. If we have a string literal longer than 32 characters we will write outside of the allocated buffer which can lead to problems.

4.3

If the return value is located just after the buffer on the stack it is susceptible to being overwritten in this case. This can allow an attacker to

change the return address to point to some harmful code, which is a huge security flaw. This harmful code could have been placed in the buffer as well by the attacker.