Below are the some brief explanations of the different values printed for c, d, and f.

c: is defined as a variable char type which stands for “character”, and can store a single character like a letter or symbol. When it is printed as %c it prints the character ‘a’, however when it is printed as %d, it is printed as the ASCII Standard location decimal number of 97.

d: is defined as a variable int type which stands for integer, and can store integer numbers. Later 108 is assigned to int d, so when printed as %d it prints the integer 108, however when printed with %c it prints the ASCII standard character of lower case ‘l’.

f: is defined as a variable double type which stands for double or floating point, and can store decimal numbers. Later 300.2 is assigned to double f. When printed with %f it prints a decimal number, some floating point numbers cannot be exactly represented as you can see by the additional digits in 300.200012. When printed with %e the double is being printed in exponential form using powers of 10.

I was actually quite surprised by a lot of what I learned by examining this program. Although we read about a lot of this information, seeing it actually in use concreted the subject. In particular, using %d to print a character and getting its ASCII number, or using %c to print an int and getting its ASCII char. It was really cool actually seeing that 300.2 was not represented precisely by float.