Question #1:

a) Software Defects:
   - On line 14, the variable “password” is used, but “password” is not defined. The variable “pass” should be used instead.
   - The "gets()" function is used to read input, which is considered unsafe as it does not check the size of the input and may lead to a buffer overflow. "fgets()" or "scanf()" can be used instead as a safer alternative. (line 11)
   - The comparison in the if statement "if (password == 1)" will always evaluate to false, as "password" is never assigned a value of 1. It should be "if (isValidPass == 1)". (line 16)
   - On line 18, printf is not declared in that scope. #include <cstdio> needs to be added at the top
   - On line 18, there is a missing semicolon that should be at the end of the line.

b) Security Flaws:
   - The password is hardcoded and stored in plaintext in the code, which is not secure. It would be better to use a cryptographic hash of the password and store that in the code, or use a password storage library. (line 3)
   - The use of the "gets()" function to read input is considered unsafe as it does not check the size of the input and may lead to a buffer overflow. An attacker could provide an oversized input to cause a buffer overflow and potentially execute arbitrary code. (line 11)

Question #2:

c) Software Defects:
   - On line 10, the function “strcat()” is used, but there is no including of “<cstring>”. #include <cstring> should be added at the top of the file.
   - On line 11, the function "strat()" is used instead of "strcat()", which will cause a compilation error. This should be corrected to "strcat()".
   - On line 10, the command line buffer is not checked for overflow, it could cause a buffer overflow issue.
   - On line 10, the command line buffer is not null-terminated which could cause unexpected behavior.

d) Security Flaws:
   - On line 8, the command line buffer is not checked for overflow, it could cause a buffer overflow issue, an attacker could provide an oversized input to cause a buffer overflow and potentially execute arbitrary code.
   - On line 7, The program does not limit the number of command line arguments, an attacker could potentially provide a large number of arguments to cause a buffer overflow or other type of attack