1.

I added this line of code to the a1ms4.c file:

void Clear() { while (getchar() != '\n'); }

and in contacts.c file, after each scanf statement, I added Clear(); that does the same thing as '%[\n]' or ' " %c" '. With this while loop, the program continues to receive input from the user until the user presses Enter; that is considered as '\n'.

scanf("%30[^\n]%\*c", Name->firstName); -> scanf("%30[^\n]", name->firstName); Clear();

scanf("%s%\*c", &yesNo); -> scanf(" %c", &yesNo); Clear();

scanf("%6[^\n]%\*c", name->middleInitial); -> scanf("%6[^\n]", &name->middleInitial); Clear();

scanf("%35[^\n]%\*c", name->lastName); -> scanf("%35[^\n]", &name->lastName); Clear();

scanf("%d%\*c", &address->streetNumber); -> scanf("%d", &address->streetNumber); Clear();

scanf("%40[^\n]%\*c", address->street); -> scanf("%40[^\n]", address->street); Clear();

scanf("%c%\*c", &yesNo); -> scanf(" %c", &yesNo); Clear();

if (yesNo == "y" || yesNo == "Y") { -> if (yesNo == 'y' || yesNo == 'Y') {

(here the characters 'y' and 'Y' were in double quotes, and it should be in single quotes.)

scanf("%d%\*c", &address->apartmentNumber); -> scanf("%d", &address->apartmentNumber); Clear();

scanf("%7[^\n]%\*c", address->postalCode); -> scanf(" %7[^\n]", address->postalCode); Clear();

scanf("%40[^\n]%\*c", address->city); -> scanf("%40[^\n]", address->city); Clear();

scanf("%c%\*c", &yesNo); -> scanf(" %c", &yesNo); Clear();

scanf("%10[^\n]%\*c", numbers->cell); -> scanf("%10[^\n]", numbers->cell); Clear();

scanf("%c%\*c", &yesNo); -> scanf(" %c", &yesNo); Clear();

if (yesNo == 'y' && yesNo == 'Y') -> if (yesNo == 'y' || yesNo == 'Y')

( with the logical operand AND &&, both of the conditions should be true in order to produce the value 1, otherwise the result would be 0 and the if condition would be false. in this situation, we only need one of the characters to be entered by the user, not both. So the logical OR operator || solves the problem.)

scanf("%10[^\n]%\*c", numbers->home); -> scanf("%10[^\n]", numbers->home); Clear();

scanf("%c%\*c", &yesNo); -> scanf(" %c", &yesNo); Clear();

scanf("%10[^\n]%\*c", numbers->business); -> scanf("%10[^\n]", numbers->business); Clear();

2.

If a struct is included within another struct, there is no need to separately allocate memory for that struct; but if the struct has a pointer to another struct, then we dynamically allocatememory for it, it is called malloc.

3.

I searched in Stack Overflow that how can the user's prompt no get corrupted when he enters a space or a special key like dot; and I found out that I can design a void function to do that for me (I explained it at the beginning of the file). The function is a void function that returns nothing and has no parameters. It includes a while loop that prevents the terminal to stop reading the input if the user presses space key or a special key.

P.S Edit: the matrix compiler gave me this error that the "Clear()" is not declared in contacts.c; so I had to move the function from a1ms4.c to contacts.c.

4.

What %\*c does in a scanf statement is that it reads a character and then ignores and discards it. It is mostly used to read up the input up to some delimiter. In our code, it means to read until \n is entered, and then discard it.

5.

Using "%[^\n]" in scanf means scan everything until a \n is entered; that basically means scan the input until the user presses Enter. There are some cases that input has spaces in between like "First Name: Amir Sabagh". In these occasions, if we use '%s', the compiler would not scan the input after the user presses Space key; the output of the previous example with '%s' will be "First Name: Amir". The solution is to replace '%[\n]' with '%s'. Another solution is to design a function to do that, like the thing I did with the code.