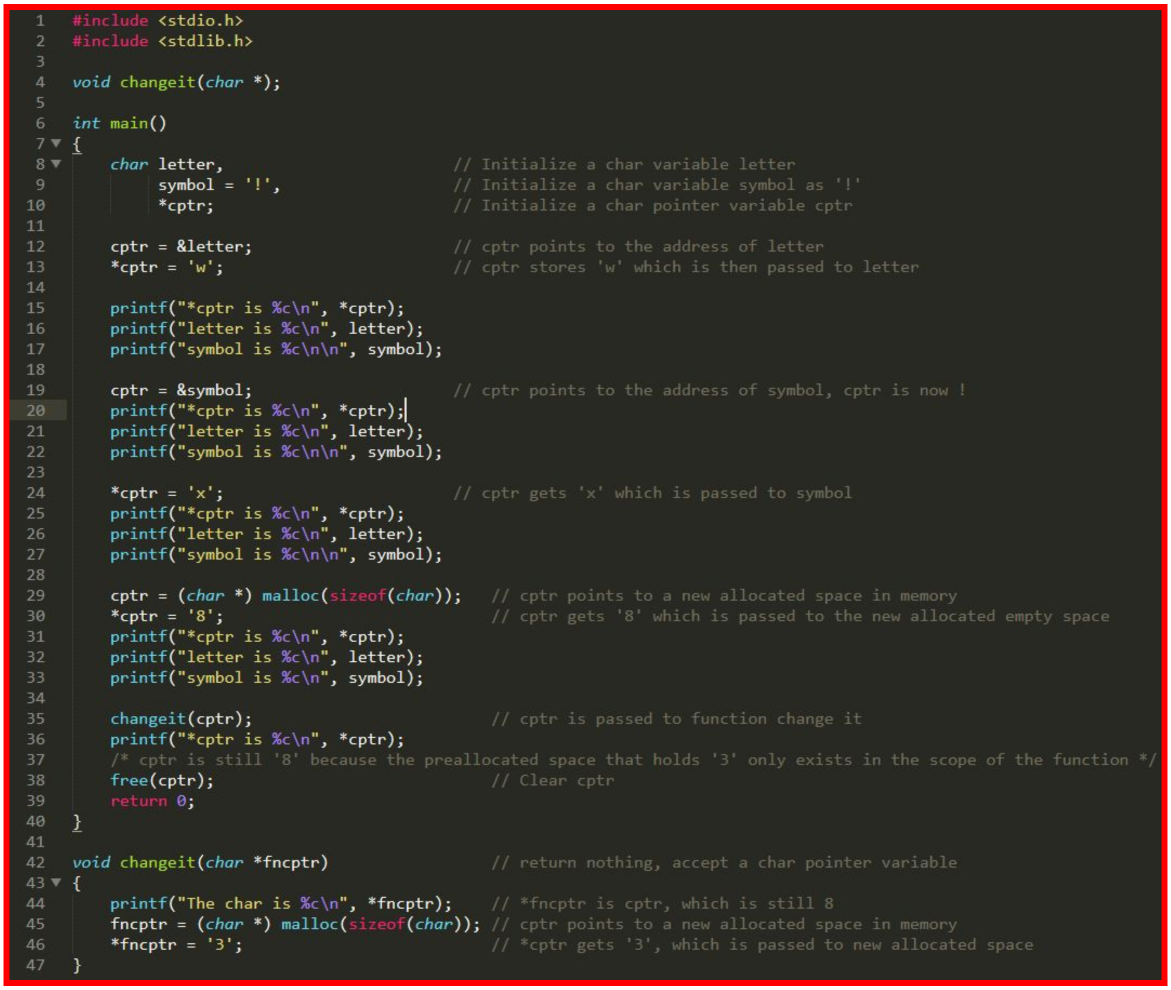
ENG EK 125 - Worksheet C Chapter 5A

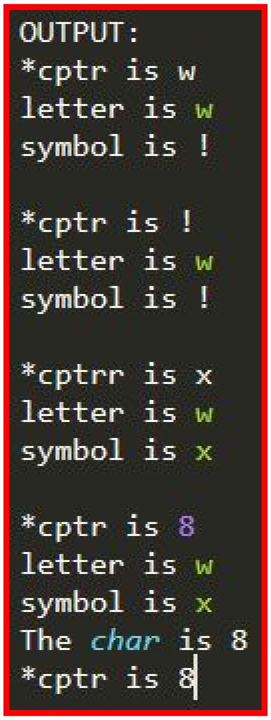
Name: Ande Chen Section: C1

1) For the following program, create a document that shows what the output would be, and also explains the output.

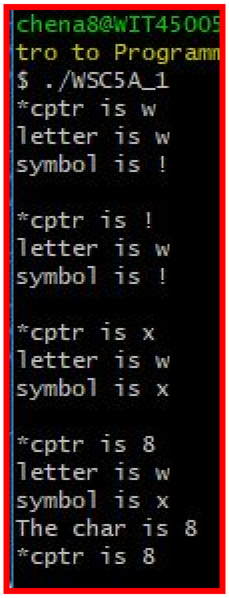
```
#include <stdio.h>

void changeit(char *);

int main()
{
    char letter,
         symbol = '!',
         *cptr;

    cptr = &letter;
    *cptr = 'w';
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    cptr = &symbol;
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    *cptr = 'x';
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    cptr = (char *) malloc(sizeof(char));
    *cptr = '8';
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n", symbol);

    changeit(cptr);
    printf("*cptr is %c\n", *cptr);
    free cptr;
    return 0;
}

void changeit(char *fncptr)
{
    printf("The char is %c\n", *fncptr);
    fncptr = (char *) malloc(sizeof(char));
    *fncptr = '3';
}
```



```c
#include <stdio.h>
#include <stdlib.h>

void changeit(char *);

int main()
{
    char letter,                        // Initialize a char variable letter
         symbol = '!',                  // Initialize a char variable symbol as '!'
         *cptr;                         // Initialize a char pointer variable cptr

    cptr = &letter;                     // cptr points to the address of letter
    *cptr = 'w';                        // cptr stores 'w' which is then passed to letter

    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    cptr = &symbol;                     // cptr points to the address of symbol, cptr is now !
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    *cptr = 'x';                        // cptr gets 'x' which is passed to symbol
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    cptr = (char *) malloc(sizeof(char));   // cptr points to a new allocated space in memory
    *cptr = '8';                            // cptr gets '8' which is passed to the new allocated empty space
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n", symbol);

    changeit(cptr);                         // cptr is passed to function change it
    printf("*cptr is %c\n", *cptr);
    /* cptr is still '8' because the preallocated space that holds '3' only exists in the scope of the function */
    free(cptr);                             // Clear cptr
    return 0;
}

void changeit(char *fncptr)                 // return nothing, accept a char pointer variable
{
    printf("The char is %c\n", *fncptr);    // *fncptr is cptr, which is still 8
    fncptr = (char *) malloc(sizeof(char)); // cptr points to a new allocated space in memory
    *fncptr = '3';                          // *cptr gets '3', which is passed to new allocated space
}
```

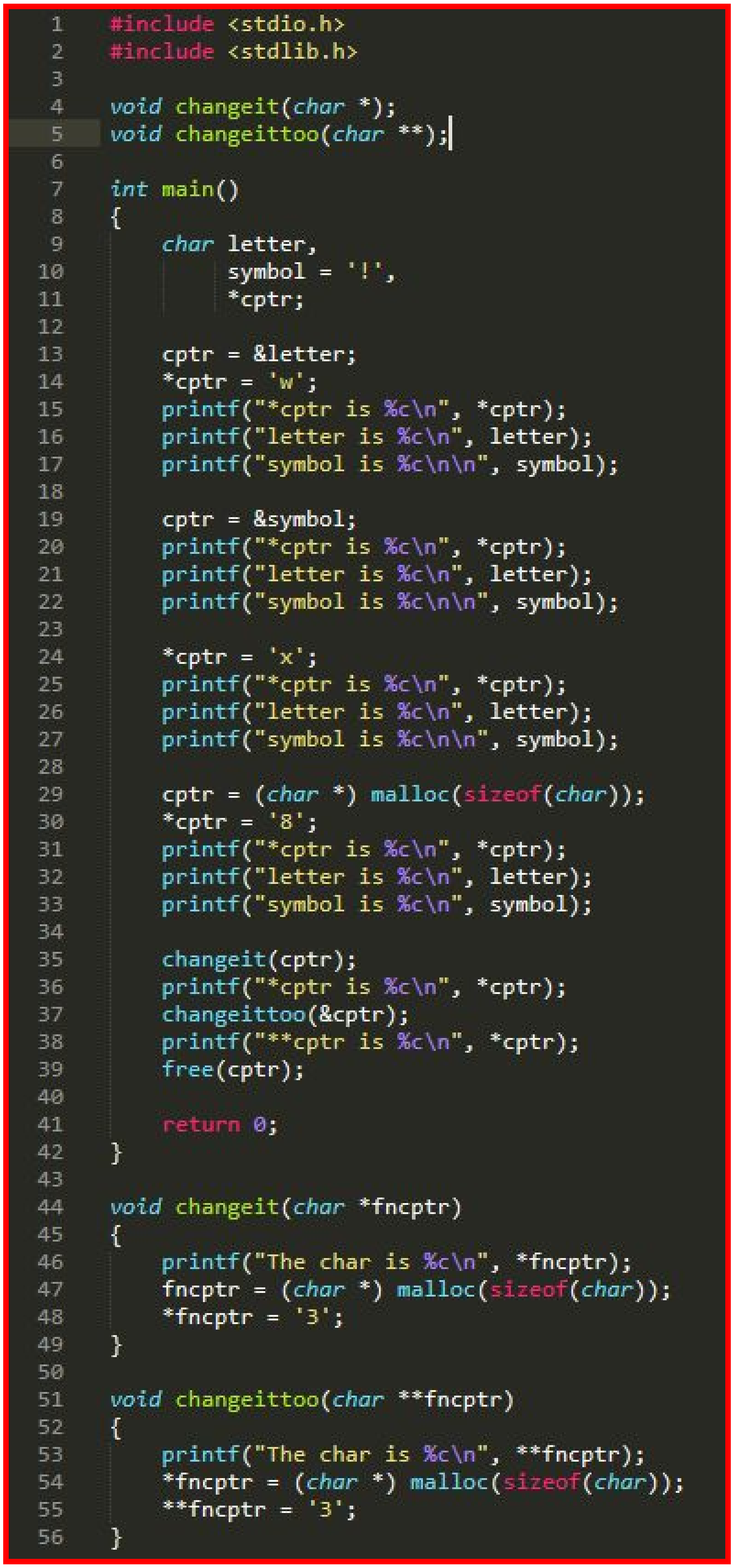


```
OUTPUT:
*cptr is w
letter is w
symbol is !

*cptr is !
letter is w
symbol is !

*cptrr is x
letter is w
symbol is x

*cptr is 8
letter is w
symbol is x
The char is 8
*cptr is 8
```

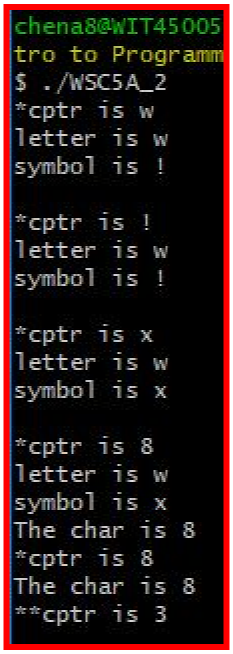


```
chena8@WIT45005
tro to Programm
$ ./WSC5A_1
*cptr is w
letter is w
symbol is !

*cptr is !
letter is w
symbol is !

*cptr is x
letter is w
symbol is x

*cptr is 8
letter is w
symbol is x
The char is 8
*cptr is 8
```

2) Add another function *changeittoo* to the above program, similar to the *changeit* function, but instead of passing cptr, pass &cptr.

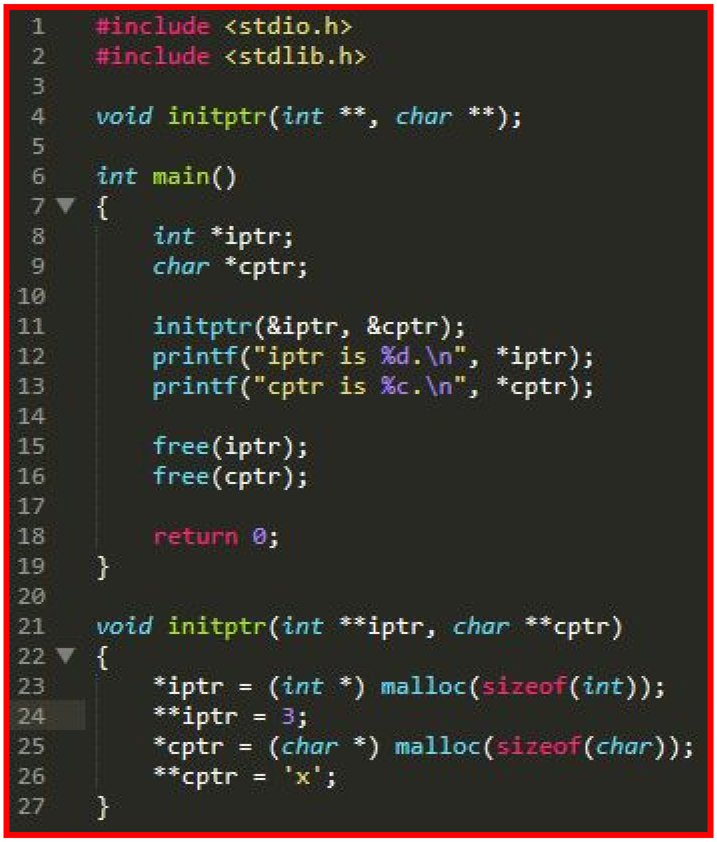


```c
#include <stdio.h>
#include <stdlib.h>

void changeit(char *);
void changeittoo(char **);

int main()
{
    char letter,
         symbol = '!',
         *cptr;

    cptr = &letter;
    *cptr = 'w';
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    cptr = &symbol;
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    *cptr = 'x';
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n\n", symbol);

    cptr = (char *) malloc(sizeof(char));
    *cptr = '8';
    printf("*cptr is %c\n", *cptr);
    printf("letter is %c\n", letter);
    printf("symbol is %c\n", symbol);

    changeit(cptr);
    printf("*cptr is %c\n", *cptr);
    changeittoo(&cptr);
    printf("**cptr is %c\n", *cptr);
    free(cptr);

    return 0;
}

void changeit(char *fncptr)
{
    printf("The char is %c\n", *fncptr);
    fncptr = (char *) malloc(sizeof(char));
    *fncptr = '3';
}

void changeittoo(char **fncptr)
{
    printf("The char is %c\n", **fncptr);
    *fncptr = (char *) malloc(sizeof(char));
    **fncptr = '3';
}
```

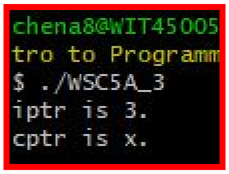


```
chena8@WIT45005
tro to Programm
$ ./WSC5A_2
*cptr is w
letter is w
symbol is !

*cptr is !
letter is w
symbol is !

*cptr is x
letter is w
symbol is x

*cptr is 8
letter is w
symbol is x
The char is 8
*cptr is 8
The char is 8
**cptr is 3
```

3) Write a program that declares two pointer variables, one that points to an integer location, and one that points to a character location. Call a function that will initialize both by dynamically allocating memory and then putting values in the allocated locations. From main, print these values. Note that you must use the call-by-reference method to initialize the pointer variables from the function.

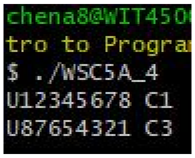


```c
#include <stdio.h>
#include <stdlib.h>

void initptr(int **, char **);

int main()
{
    int *iptr;
    char *cptr;

    initptr(&iptr, &cptr);
    printf("iptr is %d.\n", *iptr);
    printf("cptr is %c.\n", *cptr);

    free(iptr);
    free(cptr);

    return 0;
}

void initptr(int **iptr, char **cptr)
{
    *iptr = (int *) malloc(sizeof(int));
    **iptr = 3;
    *cptr = (char *) malloc(sizeof(char));
    **cptr = 'x';
}
```



```
chena8@WIT45005
tro to Programm
$ ./WSC5A_3
iptr is 3.
cptr is x.
```

4) Write a program that defines a type for a structure that stores information on a student in ENG EK 125. Declare two variables to be this structure type, and call a function to initialize both of the structure variables, using call-by-reference.



```
chena8@WIT450
tro to Progra
$ ./WSC5A_4
U12345678 C1
U87654321 C3
```