Duration: **50 minutes**
Aids Allowed: **1 - 8.5x11 sheet**

Student Number: _____

Last Name: SOLUTION

First Name: _____

Instructor: Karen Reid

---

*Do **not** turn this page until you have received the signal to start.*
(In the meantime, please fill out the identification section above, and read the instructions below *carefully*.)

This midterm test consists of 4 questions on 6 pages (including this one). *When you receive the signal to start, please make sure that your copy of the test is complete.* Extra space was left for each of the programming questions. Please indicate clearly the part of your work that should be marked.

IMPORTANT: Assume all #includes have been done for you. You do not need to perform error checking unless the question specifically asks you to.

MARKING GUIDE

1: _____/ 6

2: _____/ 5

3: _____/ 8

4: _____/ 9

TOTAL: _____/28

Good Luck!

Question 1. [6 MARKS]

Fill in the output for each print statement in the line beside it. If an error would occur, write "error", comment out the line that would cause the error, and then keep going as if the error hadn't happened. *Treat the code as one continuous program.*

```
int j = 15;
int k = 20;
k = j;
printf("j = %d,  k = %d\n",
        j, k);                          //_____j = 15, k = 20___________

int *p;
p = &k;
*p = 3;
printf("j = %d,  k = %d, *p = %d\n",
        j, k, *p);                      //_____j = 15, k = 3, *p = 3_____

char str1[10] = "Halloween";
char *ptr = "Apples";

str1[1] = 'o';
str1[6] = '\0';
printf("str1 = %s, str1 end = %s\n",  //_____str1 = Hollow, str1 end = en
        str1, &str1[7]);

//ptr[0] = 'O';
//printf("ptr = %s\n", ptr);           //_____________error_____________
strncpy(str1, ptr, 10);
printf("str1 = %s\n", str1);          //_____str1 = Apples_____________

strncpy(str1, "eve", 3);
printf("str1 = %s\n", str1);          //_____str1 = eveles_____________

struct point {
    int x;
    int y;
};
struct point a_point= {2, 3};
struct point b_point;
a_point = b_point;
b_point.x = 11;
struct point *q;
q = &a_point;
q->y = 33;
printf("a_point = (%d, %d)\n",        //______(??, 33)__x is undefined_____
    a_point.x, a_point.y);
printf("b_point = (%d, %d)\n",
        b_point.x, b_point.y);        //______(11, ??) y is undefined______
```

Question 2. [5 MARKS]

My current working directory contains the following files:

```
Makefile        message.c       message.h       printlog.c       queue.c
```

The Makefile has the following contents.

```
all : printlog

printlog : printlog.o message.o queue.o
        gcc -Wall -g -o printlog printlog.o message.o queue.o

printlog.o : printlog.c
        gcc -Wall -g -c printlog.c

message.o : message.c
        gcc -Wall -g -c message.c

queue.o : queue.c
```

Part (a) [1 MARK]

Give the target name of the rule that is run when I type `make`

"all"

Part (b) [2 MARKS]

List the files that are created when I run `make` the first time.

printlog.o, message.o, queue.o, printlog

Part (c) [2 MARKS]

If I change the file `message.c` and run `make` again, which rules in the Makefile are executed? (In other words, which gcc lines are run?)

message.o and printlog or gcc -Wall -g -c message.c, and gcc -Wall -g -o printlog printlog.o message.o queue.o)

Question 3. [8 MARKS]

Given the following struct definition:

```
struct node {
    char *str;
    struct node *next;
};
```

Part (a) [4 MARKS]

Complete the `create_node` function according to its comment. Memory allocated for the new string must be exactly the number of bytes required.

```
/* Returns a pointer to a newly created node struct. The string str is
 * is copied into memory pointed to by the str field of the new struct. The
 * pointer next is copied to the next field of the new struct.
 */
struct node *create_node(char *str, struct node *next) {
    struct node *n = malloc(sizeof(struct node));
    n->str = malloc(strlen(str) + 1);
    strncpy(n->str, str, strlen(str + 1));
    n->next = next;
    return n;
}
```

Part (b) [4 MARKS]

Complete the `freelist` function according to its comment:

```
/* De-allocates all memory allocated for the list pointed to by head.
 */
void freelist(struct node *head) {
    struct node *ptr;
    while(head != NULL) {
        ptr = head;
        head = head->next;
free(ptr->str);
        free(ptr);

    }
}
```

Question 4. [9 MARKS]

Complete the function below. You **must not** use any functions from the string library. Your function may include local integer variables, but if it include additional local string variables (other than `dest` and `newstr`) your solution will receive a maximum of 6/9.

```
/* Insert the string newstr into the string dest immediately after the first
 * occurrence of the character c.  If dest does not have the capacity
 * (given by size_dest) to hold both dest and newstr, return -1.  If the
 * character c does not occur in the string dest, return -2.  Otherwise,
 * do the insertion and return 0.
 */
int str_insert(char *dest, char *newstr, char c, int size_dest) {
    if(strlen(dest) + strlen(newstr) + 1 > size_dest) {
        return -1;
    }

    int i = 0;
    while((i < strlen(dest)) && (dest[i] != c)) {
        i++;
    }

    if(i == strlen(dest)) {
        return -2;
    } else {

        /* move the last part of dest out of the way */
        int k = strlen(dest) + strlen(newstr) -1;
        int j = strlen(dest) -1;

        for(j = strlen(dest) - 1; j > i; j--, k--) {
            dest[k] = dest[j];
        }

        i++;
        for(k = 0; k < strlen(newstr); k++) {
            dest[i + k] = newstr[k];
        }
    }
    return 0;
}
```