Assignment 10

```
#include <stdio.h>

// a. mystrcpy
char* mystrcpy(char* dest, const char* src) {
   char* ptr = dest;

   while (*src)
      *ptr++ = *src++;

   *ptr = '\0';

   return dest;
}

// b. mystrlen
int mystrlen(const char* str) {

   int len = 0;
   while (*str++)

      len++;
   return len;

}

// c. mystrcmp
int mystrcmp(const char* s1, const char* s2) {

   while (*s1 && (*s1 == *s2)) {
```

```
        s1++;
        s2++;

    }
    return *(unsigned char*)s1 - *(unsigned char*)s2;
}

// d. mystrcat
char* mystrcat(char* dest, const char* src) {

    char* ptr = dest + mystrlen(dest);

    while (*src)
        *ptr++ = *src++;
    *ptr = '\0';

    return dest;
}

// e. mystrncpy
char* mystrncpy(char* dest, const char* src, int n) {

    char* ptr = dest;
    while (n-- && *src)
        *ptr++ = *src++;

    while (n-- >= 0)
        *ptr++ = '\0';

    return dest;
}
```

```
// f. mystrupper
char* mystrupper(char* str) {

    char* ptr = str;
    while (*ptr) {

        if (*ptr >= 'a' && *ptr <= 'z')
            *ptr -= 32;
        ptr++;

    }
    return str;
}

// g. mystrlower
char* mystrlower(char* str) {

    char* ptr = str;
    while (*ptr) {

        if (*ptr >= 'A' && *ptr <= 'Z')
            *ptr += 32;

        ptr++;
    }
    return str;
}

// h. mystrrev
char* mystrrev(char* str) {
```

```
    int len = mystrlen(str);

    for (int i = 0; i < len / 2; i++) {

        char tmp = str[i];

        str[i] = str[len - i - 1];
        str[len - i - 1] = tmp;
    }

    return str;
}

// i. mystrstr
char* mystrstr(const char* haystack, const char* needle) {

    while (*haystack) {
        const char* h = haystack;
        const char* n = needle;

        while (*h && *n && *h == *n) {
            h++;
            n++;

        }

        if (!*n)

            return (char*)haystack;
        haystack++;
```

```
    }
    return NULL;
}

// j. mystrcasecmp
int mystrcasecmp(const char* s1, const char* s2) {

    while (*s1 && *s2) {
        char c1 = (*s1 >= 'A' && *s1 <= 'Z') ? *s1 + 32 : *s1;
        char c2 = (*s2 >= 'A' && *s2 <= 'Z') ? *s2 + 32 : *s2;

        if (c1 != c2)

            return c1 - c2;

        s1++;
        s2++;
    }
    return *s1 - *s2;
}

// k. mystrchr
char* mystrchr(const char* str, int c) {
    while (*str) {
        if (*str == (char)c)

            return (char*)str;

        str++;
    }
    return NULL;
```

```
}

// l. mystrrchr
char* mystrrchr(const char* str, int c) {
   const char* last = NULL;
   while (*str) {
      if (*str == (char)c)

         last = str;

      str++;
   }
   return (char*)last;
}

// m. mystrncmp
int mystrncmp(const char* s1, const char* s2, int n) {
   while (n-- && *s1 && *s2) {

      if (*s1 != *s2)

         return *(unsigned char*)s1 - *(unsigned char*)s2;
      s1++;
      s2++;
   }
   return 0;
}

// n. mystrnstr
char* mystrnstr(const char* haystack, const char* needle, int len) {
   int needle_len = mystrlen(needle);
```

```
    for (int i = 0; i <= len - needle_len; i++) {

        if (!mystrncmp(haystack + i, needle, needle_len))

            return (char*)(haystack + i);
    }
    return NULL;
}

// o. mystrncat
char* mystrncat(char* dest, const char* src, int n) {
    char* ptr = dest + mystrlen(dest);
    while (n-- && *src)

        *ptr++ = *src++;

    *ptr = '\0';
    return dest;
}

// p. mystrncasecmp
int mystrncasecmp(const char* s1, const char* s2, int n) {
    while (n-- && *s1 && *s2) {
        char c1 = (*s1 >= 'A' && *s1 <= 'Z') ? *s1 + 32 : *s1;
        char c2 = (*s2 >= 'A' && *s2 <= 'Z') ? *s2 + 32 : *s2;
        if (c1 != c2)

            return c1 - c2;
        s1++;
        s2++;
    }
```

```
    return 0;
}
// Sample main function
int main() {
    char str1[100] = "Manish";
    char str2[100] = "Mahale";

    printf("mystrcpy: %s\n", mystrcpy(str1, str2));
    // mystrcpy
    printf("\n----------------------------------------------------------------\n\n");

    printf("mystrlen: %d\n", mystrlen("Manish Mahale"));
    // mystrlen
    printf("\n----------------------------------------------------------------\n\n");

    printf("mystrcmp: %d\n", mystrcmp("Manish", "Mahale"));
    // mystrcmp
    printf("\n----------------------------------------------------------------\n\n");

    printf("mystrcat: %s\n", mystrcat(str1,  "123"));
    // mystrcat
    printf("\n----------------------------------------------------------------\n\n");

    printf("mystrncpy: %s\n", mystrncpy(str1, "abcdef", 3));
    // mystrncpy
    printf("\n----------------------------------------------------------------\n\n");
    printf("mystrupper: %s\n", mystrupper(str1));
    // mystrupper
    printf("\n----------------------------------------------------------------\n\n");

    printf("mystrlower: %s\n", mystrlower(str1));
```

```
// mystrlower
printf("\n-----------------------------------------------------------------\n\n");

printf("mystrrev: %s\n", mystrrev(str1));
// mystrrev
printf("\n-----------------------------------------------------------------\n\n");

printf("mystrstr: %s\n", mystrstr("Manish Mahale", "Mahale"));
// mystrstr
printf("\n-----------------------------------------------------------------\n\n");

printf("mystrcasecmp: %d\n", mystrcasecmp("MANISH",  "manish"));
// mystrcasecmp
printf("\n-----------------------------------------------------------------\n\n");

printf("mystrchr: %s\n", mystrchr("Manish Mahale", 'a'));
// mystrchr
printf("\n-----------------------------------------------------------------\n\n");

printf("mystrrchr: %s\n", mystrrchr("Manish Mahale", 'a'));
// mystrrchr
printf("\n-----------------------------------------------------------------\n\n");

printf("mystrncmp: %d\n", mystrncmp("Manish", "Mahale",3));
// mystrncmp
printf("\n-----------------------------------------------------------------\n\n");

printf("mystrnstr: %s\n", mystrnstr("Manish Mahale", "ale", 12));
// mystrnstr
printf("\n-----------------------------------------------------------------\n\n");
```

```
    printf("mystrncat: %s\n", mystrncat(str1, "XYZ", 2));
    // mystrncat
    printf("\n----------------------------------------------------------------\n\n");

    printf("mystrncasecmp: %d\n", mystrncasecmp("Manish","manIsh", 6));
    // mystrncasecmp
    printf("\n----------------------------------------------------------------\n\n");

    return 0;
}
```