```c
// Created by hfwei on 2022/11/25.

#include <stdio.h>
#include <string.h>

#define LEN 10

void Swap(char **left, char **right);
void PrintStrs(const char *str[], int len);
void SelectionSort(char *str[], int len);

int main() {
  const char *musicians[LEN] = {
      "Luo Dayou",
      "Cui Jian",
      "Dou Wei",
      "Zhang Chu",
      "Li Zhi",
      "Wan Qing",
      "WuTiaoRen",
      "ZuoXiao",
      "He Mage",
      "He Yong",
  };

  PrintStrs(musicians, LEN);
  SelectionSort(musicians, LEN);
  PrintStrs(musicians, LEN);
}

void PrintStrs(const char *str[], int len) {
  printf("\n");
  for (int i = 0; i < len; i++) {
    printf("%s\n", str[i]);
  }
  printf("\n");
}

// arr: the (copy of the) address of the first element of the `numbers` array
void SelectionSort(char *str[], int len) {
  for (int i = 0; i < len; i++) {
    // find the minimum of musicians[i .. len - 1]
    const char *min = str[i];
    int min_index = i;

    for (int j = i + 1; j < len; j++) {
      if (strcmp(min, str[j]) > 0) {
        min = str[j];
        min_index = j;
      }
    }

    // swap str[i] and str[min_index]
    Swap(str + i, str + min_index);
  }
}

void Swap(char **left, char **right) {
  char *temp = *left;
  *left = *right;
  *right = temp;
}
```

```c
//
// Created by hfwei on 2022/11/25.
//

#include <stdio.h>

/**
 * @brief Compares the string pointed to by s1 to the string pointed to by s2
 * @param s1 the first string
 * @param s2 the second string
 * @return > 0 if s1 > s2; 0 if s1 = s2; < 0 if s1 < s2.
 */
int StrCmp(const char *s1, const char *s2);
int StrCmpStd(const char *s1, const char *s2);

/**
 * @brief Compare no more than n characters
 *        (characters that follow '\0' are not compared)
 *        from the array pointed to by s1 to the array pointed to by s2.
 * @param s1
 * @param s2
 * @param n
 * @return
 */
int StrNCmp(const char *s1, const char *s2, int n);
int StrNCmpStd(const char *s1, const char *s2, int n);

int main() {
  const char *str1 = "hi, C";
  const char *str2 = "hi, c";

  printf("StrCmp(\"%s\", \"%s\") = %d\n",
         str1, str2, StrCmp(str1, str2));
  printf("StrCmpStd(\"%s\", \"%s\") = %d\n",
         str1, str2, StrCmpStd(str1, str2));

  int n = 2;
  printf("StrNCmp(\"%s\", \"%s\", %d) = %d\n",
         str1, str2, n, StrNCmp(str1, str2, n));

  return 0;
}

int StrCmp(const char *s1, const char *s2) {
  while (*s1 == *s2 && (*s1 != '\0' && *s2 != '\0')) {
    s1++;
    s2++;
  }

  // four cases here
  if (*s1 == '\0' && *s2 == '\0') {
    return 0;
  }

  return (*(const unsigned char *) s1)
             < (*(const unsigned char *) s2) ? -1 : 1;
}

int StrCmpStd(const char *s1, const char *s2) {
  for (; *s1 == *s2; s1++, s2++) {
    if (*s1 == '\0') {
      return 0;
    }
  }

  return (*(const unsigned char *) s1)
             < (*(const unsigned char *) s2) ? -1 : 1;
}

int StrNCmp(const char *s1, const char *s2, int n) {
  for (int i = 0; i < n; i++) {
    if (*s1 != *s2) {
      return (*(const unsigned char *) s1)
                 < (*(const unsigned char *) s2) ? -1 : 1;
    } else {  // *s1 = *s2
      if (*s1 == '\0') {
        return 0;
      } else {
        s1++;
        s2++;
      }
    }
  }

  return 0;
}

int StrNCmpStd(const char *s1, const char *s2, int n) {
  for (; 0 < n; n--, s1++, s2++) {
    if (*s1 != *s2) {
      return (*(const unsigned char *) s1)
                 < (*(const unsigned char *) s2) ? -1 : 1;
    } else if (*s1 == '\0') {
      return 0;
    }
  }

  return 0;
}
```

```c
//
// Created by hfwei on 2022/11/25.
//

#include <string.h>
#include <stdio.h>

/**
 * @brief Copy the string pointed to by src (including the terminating null character)
 *  into the array pointed to by dest.
 *  If copying takes place between objects that overlap,
 *  the behavior is undefined.
 *
 *  We assume that there is enough room in dest for storing src.
 *  Otherwise, it is an undefined behavior.
 *
 * @param dest
 * @param src
 */
void StrCpy(char *dest, const char *src);
void StrCpy1(char *dest, const char *src);
void StrCpy2(char *dest, const char *src);
void StrCpy3(char *dest, const char *src);
void StrCpy4(char *dest, const char *src);
void StrCpy5(char *dest, const char *src);
char *StrCpyStd(char *dest, const char *src);

int main() {
  const char *src = "Hello World";
  char dest[strlen(src) + 1];

  // StrCpy(dest, src);
  // printf("dest = %s\n", dest);

  strcpy_s(dest, sizeof dest / sizeof dest[0], src);
  printf("dest = %s\n", dest);

  return 0;
}

void StrCpy(char *dest, const char *src) {
  int i = 0;
  while (src[i] != '\0') {
    dest[i] = src[i];
    i++;
  }

  dest[i] = '\0';
}

void StrCpy1(char *dest, const char *src) {
  int i = 0;
  while ((dest[i] = src[i]) != '\0') {
    i++;
  }
}

void StrCpy2(char *dest, const char *src) {
  int i = 0;
  while ((*(dest + i) = *(src + i)) != '\0') {
    i++;
  }
}

void StrCpy3(char *dest, const char *src) {
  while ((*dest = *src) != '\0') {
    src++;
    dest++;
  }
}

void StrCpy4(char *dest, const char *src) {
  while ((*dest++ = *src++) != '\0');
}

// NOT recommended!
void StrCpy5(char *dest, const char *src) {
  while ((*dest++ = *src++));
}

char *StrCpyStd(char *dest, const char *src) {
  for (char *s = dest; (*s++ = *src++) != '\0';);
  return dest;
}
```

```c
//
// Created by hfwei on 2022/11/25.
//

#include <stdio.h>

int StrLen(const char *str);
size_t StrLenStd(const char *str);

int main() {
  char msg[20] = "Hello World!";
  msg[0] = 'N';
  printf("%s\n", msg);

  // create an anonymous array of `char`
  // that may be stored in read-only memory
  char *ptr_msg = "Hello World!";
  // undefined behavior
  // interpreted by signal 11: SIGSEGV
  // SIG: Signal; SEGV: segmentation violation
  *ptr_msg = 'N';
  // ptr_msg[0] = 'N';
  printf("%s\n", msg);

  printf("StrLen(%s) = %d\n", msg, StrLen(msg));
  printf("StrLenStd(%s) = %d\n", msg, StrLen(msg));

  return 0;
}

int StrLen(const char *str) {
  int len = 0;
  while (str[len] != '\0') {
    len++;
  }
  // printf("%d", '\0');

  return len;
}

size_t StrLenStd(const char *str) {
  const char *sc;
  for (sc = str; *sc != '\0'; sc++);

  return (sc - str);
}
```