```c
//
// file: strcpy.c
// 7 versions of strcpy
// Created by hfwei on 2022/11/29.
//
// C Operator Precedence: https://en.cppreference.com/w/c/language/operator_precedence#:~:text=C%20Operator%20Precedence%20%20%20%20Precedence%20,union%20member%20access%20%2028%20more%20rows%20
//

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

/**
 * @brief We assume that there is enough room for storing src.
 *  Otherwise, it is an undefined behavior.
 *
 *  If copying takes place between objects that overlap,
 *  then behavior is undefined.
 *
 *  In Docs:
 *  (1) The behavior is undefined if the dest array is not large enough.
 *  (2) The behavior is undefined if the strings overlap.
 *  (3) The behavior is undefined if either dest is not a pointer to a character array
 *      or src is not a pointer to a null-terminated byte string.
 *
 * @param dest may NOT be null-terminated
 * @param src  must be null-terminated
 */
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";
  // VLA (Do not use it; it is optional since C11)
  // char dest[strlen(src) + 1];
  char *dest = malloc(strlen(src) + 1);

  StrCpy4(dest, src);
  strlen(dest);
  printf("dest = %s\n", dest);

  strlen(StrCpyStd(dest, src));

  return 0;
```

```c
}

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;
  // dest[i] : *(dest + i)
  while ((*(dest + i) = *(src + i)) != '\0') {
    i++;
  }
}

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

  printf("%s\n", src);
}

// See C Operator Precedence: https://en.cppreference.com/w/c/language/operator_precedence#:~:text=C%20Operator%20Precedence%20%20%20%20Precedence%20,union%20member%20access%20%2028%20more%20rows%20
// Visualization: https://pythontutor.com/render.html#code=%23include%20%3Cstring.h%3E%0A%23include%20%3Cstdio.h%3E%0A%23include%20%3Cstdlib.h%3E%0A%0Avoid%20StrCpy4%28char%20*dest,%20const%20char%20*src%29%3B%0A%0Aint%20main%28%29%20%7B%0A%20%20const%20char%20*src%20%3D%20%22Hello%20World%22%3B%0A%20%20char%20*dest%20%3D%20malloc%28strlen%28src%29%20%2B%201%29%3B%0A%0A%20%20StrCpy4%28dest,%20src%29%3B%0A%20%20printf%28%22dest%20%3D%20%25s%5Cn%22,%20dest%29%3B%0A%0A%20%20return%200%3B%0A%7D%0A%0A%0Avoid%20StrCpy4%28char%20*dest,%20const%20char%20*src%29%20%7B%0A%20%20//%20dest%2B%2B%3A%20dest,%20dest%20%3D%20dest%20%2B%201%0A%20%20//%20*dest%2B%2B%3A%20*dest,%20not%20*%28dest%20%2B%201%29%0A%20%20while%20%28%0A%20%20%20%20%28*dest%2B%2B%20%0A%20%20%20%20%20%20%3D%20*src%2B%2B%29%20%0A%20%20%20%20%20%20%20%20%20%20%20%20!%3D%20'%5C0'%29%3B%0A%7D&cppShowMemAddrs=true&cumulative=true&curInstr=0&heapPrimitives=nevernest&mode=display&
```

```c
// origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false
// Tricky difference between StrCpy3: src, dest beyond '\0'
// You SHOULD be able to understand this!!!
void StrCpy4(char *dest, const char *src) {
  // dest++: dest, dest = dest + 1
  // *dest++: *dest, not *(dest + 1)
  while ((*dest++ = *src++) != '\0');

  printf("%s\n", src);
}

// NOT Recommended!
// See ASCII Chart: https://en.cppreference.com/w/c/language/ascii
void StrCpy5(char *dest, const char *src) {
  // '\0': null character (NUL), 0
  // `\0' is not NULL
  while ((*dest++ = *src++));
}

// See https://en.cppreference.com/w/c/string/byte/strcpy
char *StrCpyStd(char *dest, const char *src) {
  for (char *s = dest; (*s++ = *src++) != '\0';);
  return dest;
}
```