C Programming
Day15.B

2017.11.03

strcpy(), pointer manipulation

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

int main(void) {
  char S[30] = "AAA BBB CCC";
  char *p, *q;
  int i;

  printf("sizeof(S)= %ld \n", sizeof(S));
  printf("strlen(S)= %ld \n", strlen(S));

  ///// Method 1 ////////////////////////////////
  ///// S = "GGG HHH"; // Not Working

  ///// Method 2 ////////////////////////////////
  p = "GGG HHH";

  for (i=0; i<=strlen(p); ++i) S[i] = p[i];
  printf("S= %s\n", S);

  ///// Method 3 ////////////////////////////////
  p = "GGG HHH";

  for (i=0; i<=strlen(p); ++i) *(S+i) = *(p+i);
  printf("S= %s\n", S);

  ///// Method 4 ////////////////////////////////
  p = "GGG HHH";
  q = S;
  while (*p) *q++ = *p++; *q = 0;
  printf("S= %s\n", S);

  ///// Method 5 ////////////////////////////////
  p = "GGG HHH";
  q = S;
  for (i=0; i<=strlen(p); ++i) *q++ = *p++;
  printf("S= %s\n", S);

  ///// Method 6 ////////////////////////////////
  ///// while (*p) *S++ = *p++; // Not Working

  ///// Method 7 ////////////////////////////////
  strcpy(S, "GGG HHH");
  printf("S= %s\n", S);

}
```

Pointers with ++ and −− (1)

x = * (p ++);	x = *p++;	x = * (++ p);	x = *++p;
x = * (p −−);	x = *p−−;	x = * (−− p);	x = *−−p;

Access First

x = * (p ++);

x = * (p −−);

Update First

x = * (++ p);

x = * (−− p);

Update Next

x = * (p ++);

x = * (p −−);

Access Next

x = * (++ p);

x = * (−− p);

Precedence	Operator	Description	Associativity
1	`++ --`	Suffix/postfix increment and decrement	Left-to-right
	`()`	Function call	
	`[]`	Array subscripting	
	`.`	Structure and union member access	
	`->`	Structure and union member access through pointer	
	`(type){list}`	Compound literal(C99)	
2	`++ --`	Prefix increment and decrement	Right-to-left
	`+ -`	Unary plus and minus	
	`! ~`	Logical NOT and bitwise NOT	
	`(type)`	Type cast	
	`*`	Indirection (dereference)	
	`&`	Address-of	
	`sizeof`	Size-of[note 1]	
	`_Alignof`	Alignment requirement(C11)	

http://en.cppreference.com/w/c/language/operator_precedence

Pointers with ++ and −− (2)

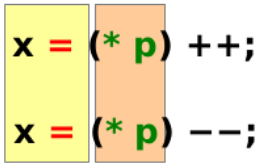
x = (* p) ++;

x = (* p) −−;

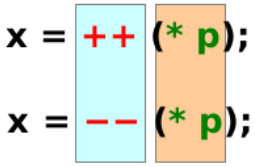
x = ++ (* p); x = ++*p;

x = −− (* p); x = −−*p;

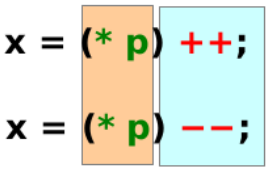
Access First



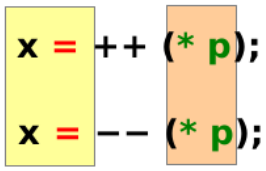
Update First



Update Next



Access Next



Pre and Post Increment / Decrement

```
v = *p++;

   v = *p      (access first)
   p = p+1     (increment later) (pointer increment)

v = (*p)++;

   v = *p      (access first)
   *p = *p+1   (increment later) (value increment)

v = *++p;

   p = p+1     (increment first) (pointer increment)
   v = *p      (access later)

v = ++*p;

   *p = *p+1   (increment first) (value increment)
   v = *p      (access later)
```

```c
#include <stdio.h>
#include <string.h>
#define SIZE 30

int main(void) {
  char  S[30];
  char  T[30];
  char *p;
  int   i;
  int   C[10];

  printf("Hello, world!\n");
  sprintf(S, "Hello, world!\n");

  printf("S= %s\n", S);

  p = s; i=0;
  while (*p)
    printf("S[%d]= %c\n", i++, *(p++));


  strcpy(s, "");
  for (i=0; i<10; ++i) {
    sprintf(T, " %d", i);
    strcat(S, T);
  }

  printf("s= %s\n", s);

  sscanf(s, "%d%d%d%d%d%d%d%d%d%d",
         C+0,C+1,C+2,C+3,C+4,C+5,C+6,C+7,C+8,C+9);

  for (i=0; i<10; ++i) {
    printf("C[%d] = %d \n", i, C[i]);
  }

}
```

```
Hello, world!
S= Hello, world!

S[0]= H
S[1]= e
S[2]= l
S[3]= l
S[4]= o
S[5]= ,
S[6]=
S[7]= w
S[8]= o
S[9]= r
S[10]= l
S[11]= d
S[12]= !
S[13]=

S=  0 1 2 3 4 5 6 7 8 9
C[0] = 0
C[1] = 1
C[2] = 2
C[3] = 3
C[4] = 4
C[5] = 5
C[6] = 6
C[7] = 7
C[8] = 8
C[9] = 9
```

```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>

#define SIZE 30

int main(void) {
  char  S[30];
  char  T[30];
  char *p;
  int   i;
  int   C[10];

  printf("Hello, world!\n");
  sprintf(S, "Hello, world!\n");

  printf("S= %s\n", S);

  p = S; i=0;
  while (*p)
    printf("S[%d]= %c\n", i++, *(p++));


  strcpy(S, "");
  for (i=0; i<10; ++i) {
    sprintf(T, " %d", i);
    strcat(S, T);
  }

  printf("S= %s\n", S);

  printf("-------------------------\n");
  p= S; i= 0;
  while (*p) {
    sscanf(p, "%d", C+i++);
    while (isspace(*p)) p++;
    while (isdigit(*p)) p++;
  }
  printf("\n");

  for (i=0; i<10; ++i) {
    printf("C[%d] = %d \n", i, C[i]);
  }

}
```

```
printf("------------------------\n");
p= S; i= 0;
while (*p) {
  sscanf(p, "%d", C+i++);
  while (isspace(*p)) p++;
  while (isdigit(*p)) p++;
}
printf("\n");
```

skip space
skip numbers

```
S= 0 1 2 3 4 5 6 7 8 9
------------------------
```

C+ i++

& C[i]

i++;