C Programming
Day15.B

2017.11.24

strcpy(), pointer manipulation
strchr(), strpbrk(), strspn(), strtok()
pre/post-inc/dec + dereferencing

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 −− (1)

③ `x = * (p ++);`	⑪ `x = *p++;`	① `x = * (++ p);`	⑨ `x = *++p;`
`x = * (p −−);`	`x = *p−−;`	`x = * (−− p);`	`x = *−−p;`
Access First: `x = * (p ++);`		Update First: `x = * (++ p);`	
		`x = * (−− p);`	
Update Next: `x = * (p ++);`		Access Next: `x = * (++ p);`	
`x = * (p −−);`		`x = * (−− p);`	

(1) *(++p) *(- - p)

(9) *++p * - - p

(3) *(p++) *(p - -)

(11) *p++ *p - -

(14) *p++ *p - -

(15) *p++ *p - -

(5) ++(*p) - - (*p)

(8) ++(*p) - - (*p)

(13) ++*p - - *p

(11) ++*p - - *p

(4) (*p)++ (*p) - -

(7) (*p)++ (*p) - -

Pointers with ++ and −− (2)

(6) (5) (13)

```
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);
```


(1) *(++p) *(- - p) (9) *++p * - - p

(3) *(p++) *(p - -) (11) *p++ *p - - (14) *p++ *p - - (15) *p++ *p - -

(5) ++(*p) - - (*p) (8) ++(*p) - - (*p) (13) ++*p - - *p (11) ++*p - - *p

(6) (*p)++ (*p) - - (7) (*p)++ (*p) - -

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)
```

(1)
```
*(++p)
*( - - p)
```

(2)
```
(++p)*
( - - p)*
```

(3)
```
*(p++)
*(p - - )
```

(4)
```
(p++)*
(p - - )*
```

(5)
```
++(*p)
 - - (*p)
```

(6)
```
(*p)++
(*p) - -
```

(7)
```
(*p)++
(*p) - -
```

(8)
```
++(*p)
 - - (*p)
```

(9)
```
*++p
* - - p
```

(10)
```
++p*
 - - p*
```

(11)
```
*p++
*p - -
```

(12)
```
p++*
p - - *
```

(13)
```
++*p
 - - *p
```

(14)
```
*p++
*p - -
```

(15)
```
*p++
*p - -
```

(16)
```
++*p
 - - *p
```

(1) *(++p)
*(- - p)

(2) ~~(++p)*~~
~~(- - p)*~~

(3) *(p++)
*(p - -)

(4) ~~(p++)*~~
~~(p - -)*~~

(5) ++(*p)
- - (*p)

(6) (*p)++
(*p) - -

(7) (*p)++
(*p) - -

(8) ++(*p)
- - (*p)

(9) *++p
* - - p

(10) ~~++p*~~
~~- - p*~~

(11) *p++
*p - -

(12) ~~p++*~~
~~p - - *~~

(13) ++*p
- - *p

(14) *p++
*p - -

(15) *p++
*p - -

(11) ++*p
- - *p

(1) *(++p) *(- - p)

(9) *++p * - - p

(3) *(p++) *(p - -)

(11) *p++ *p - -

(14) *p++ *p - -

(15) *p++ *p - -

(5) ++(*p) - - (*p)

(8) ++(*p) - - (*p)

(13) ++*p - - *p

(11) ++*p - - *p

(4) (*p)++ (*p) - -

(7) (*p)++ (*p) - -

(1) *(++p) *(- - p)

(9) *++p * - - p

(3) *(p++) *(p - -)

(11) *p++ *p - -

(14)

(15)

(5) ++(*p) - - (*p)

(8)

(13) ++*p - - *p

(11)

(4) (*p)++ (*p) - -

(7)

(1) (9) *++p
* - - p

(3) (11) *p++
*p - - (14) (15)

(5) (8) (13) ++*p
- - *p (11)

(4) (*p)++
(*p) - - (7)

```c
#include <stdio.h>

void pr(int *p, int x, char *s) {
  printf("p= %p *p= %d x= %d  %s\n", p, *p, x, s);
}

int main(void) {
  int A[] = {111, 222, 333, 444};
  int *p;
  int x = 0;

  printf("&A[0]= %p  A[0]= %d \n", &A[0], A[0]);
  printf("&A[1]= %p  A[1]= %d \n", &A[1], A[1]);
  printf("&A[2]= %p  A[2]= %d \n", &A[2], A[2]);
  printf("&A[3]= %p  A[3]= %d \n", &A[3], A[3]);

  printf("--------------------------\n");
  p = A+1;  pr(p, x, "");

  printf("--------------------------\n");
  p= A+1;  x = *p++;  pr(p, x, "x= *p++");
  p= A+1;  x = *p--;  pr(p, x, "x= *p--");

  printf("--------------------------\n");
  p= A+1;  x = *++p;    pr(p, x, "x= *++p");
  p= A+1;  x = *--p;    pr(p, x, "x= *--p");

  printf("--------------------------\n");
  p= A+1;  x = (*p)++;  pr(p, x, "x= (*p)++");
  p= A+1;  x = (*p)--;  pr(p, x, "x= (*p)--");

  printf("--------------------------\n");
  p= A+1;  x = ++*p;    pr(p, x, "x= ++*p");
  p= A+1;  x = --*p;    pr(p, x, "x= --*p");

}
```

```
&A[0]= 0x7ffcb8b20920  A[0]= 111
&A[1]= 0x7ffcb8b20924  A[1]= 222
&A[2]= 0x7ffcb8b20928  A[2]= 333
&A[3]= 0x7ffcb8b2092c  A[3]= 444
---------------------------
p= 0x7ffcb8b20924 *p= 222 x= 0
---------------------------
p= 0x7ffcb8b20928 *p= 333 x= 222  x= *p++
p= 0x7ffcb8b20920 *p= 111 x= 222  x= *p--
---------------------------
p= 0x7ffcb8b20928 *p= 333 x= 333  x= *++p
p= 0x7ffcb8b20920 *p= 111 x= 111  x= *--p
---------------------------
p= 0x7ffcb8b20924 *p= 223 x= 222  x= (*p)++
p= 0x7ffcb8b20924 *p= 222 x= 223  x= (*p)--
---------------------------
p= 0x7ffcb8b20924 *p= 223 x= 223  x= ++*p
p= 0x7ffcb8b20924 *p= 222 x= 222  x= --*p
```

```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);

}
```

```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
------------------------
```

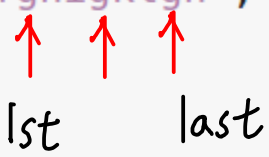
C+ i++

& C[i]

i++;

strchr()

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

int main(void) {

  //          01234567890123 // 14 char's + 1 null
  char *s = "abcdefghigklgn";
  char *p;
  char *q;
  int   i;
  int   len;


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

  p= strchr(s, 'g');
  q= strrchr(s, 'g');

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

  printf("strlen(s)= %ld \n", strlen(s));
  len = strlen(s);

  //  s[i] = *(s + i)
  // &s[i] =  (s + i)

  for (i=0; i<len; ++i)
    printf("s[%2d]= %c s+%2d= %p \n", i, s[i], i, s+i);

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


}
```



reverse

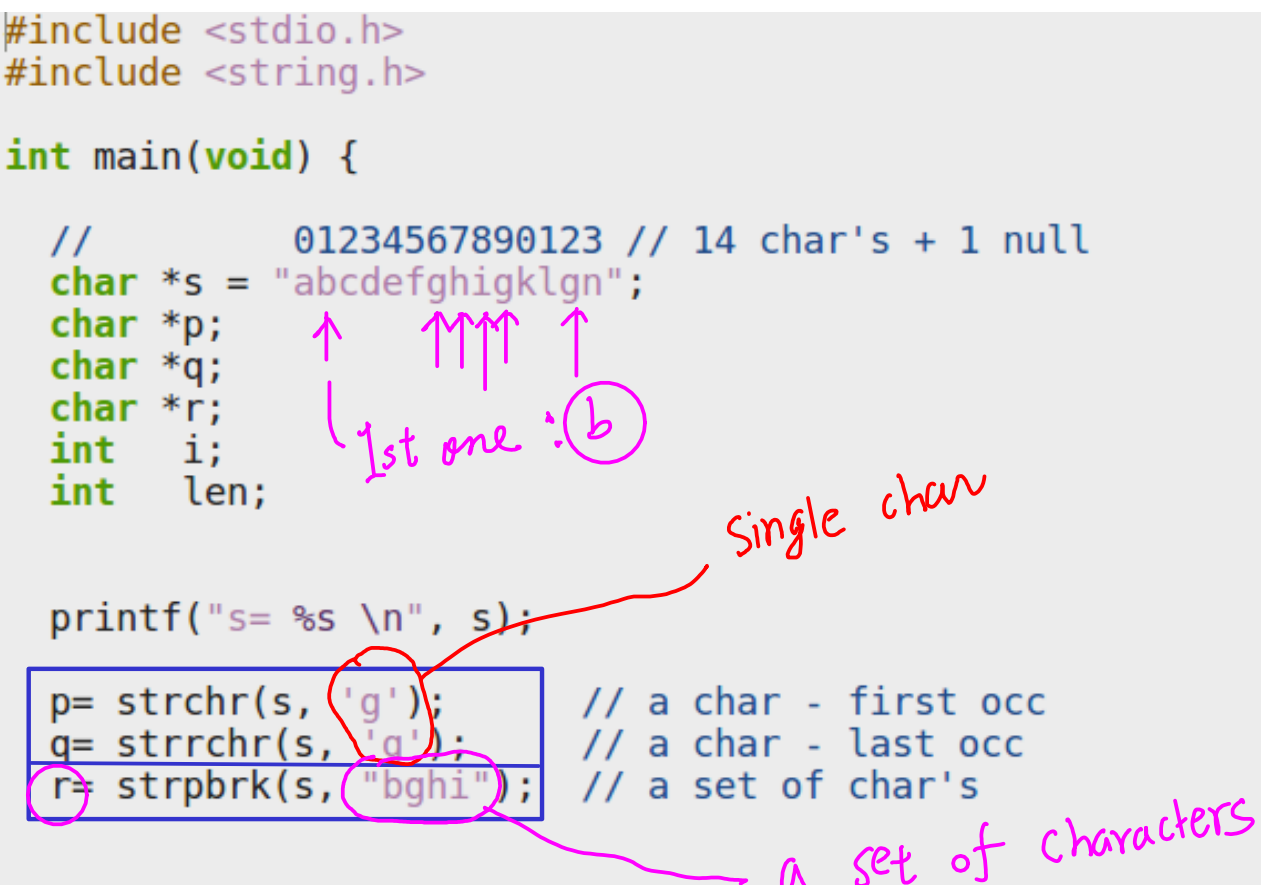
first g

```
s= abcdefghigklgn
s= abcdefghigklgn
p= ghigklgn
q= gn
strlen(s)= 14
s[ 0]= a s+ 0= 0x4007e4
s[ 1]= b s+ 1= 0x4007e5
s[ 2]= c s+ 2= 0x4007e6
s[ 3]= d s+ 3= 0x4007e7
s[ 4]= e s+ 4= 0x4007e8
s[ 5]= f s+ 5= 0x4007e9
s[ 6]= g s+ 6= 0x4007ea
s[ 7]= h s+ 7= 0x4007eb
s[ 8]= i s+ 8= 0x4007ec
s[ 9]= g s+ 9= 0x4007ed
s[10]= k s+10= 0x4007ee
s[11]= l s+11= 0x4007ef
s[12]= g s+12= 0x4007f0
s[13]= n s+13= 0x4007f1
s= 0x4007e4
p= 0x4007ea
q= 0x4007f0
```

last g

strpbrk()

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

int main(void) {

  //          01234567890123 // 14 char's + 1 null
  char *s = "abcdefghigklgn";
  char *p;
  char *q;
  char *r;
  int   i;
  int   len;


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

  p= strchr(s, 'g');       // a char - first occ
  q= strrchr(s, 'g');      // a char - last occ
  r= strpbrk(s, "bghi");   // a set of char's


  printf("s= %s \n", s);
  printf("p= %s \n", p);
  printf("q= %s \n", q);
  printf("r= %s \n", r);

  printf("strlen(s)= %ld \n", strlen(s));
  len = strlen(s);

  //  s[i] = *(s + i)
  // &s[i] =  (s + i)

  for (i=0; i<len; ++i)
    printf("s[%2d]= %c s+%2d= %p \n", i, s[i], i, s+i);

  printf("s= %p \n", s);
  printf("p= %p \n", p);
  printf("q= %p \n", q);
  printf("r= %p \n", r);


}
```



```
s= abcdefghigklgn
s= abcdefghigklgn
p= ghigklgn
q= gn
r= bcdefghigklgn
strlen(s)= 14
s[ 0]= a s+ 0= 0x400864
s[ 1]= b s+ 1= 0x400865
s[ 2]= c s+ 2= 0x400866
s[ 3]= d s+ 3= 0x400867
s[ 4]= e s+ 4= 0x400868
s[ 5]= f s+ 5= 0x400869
s[ 6]= g s+ 6= 0x40086a
s[ 7]= h s+ 7= 0x40086b
s[ 8]= i s+ 8= 0x40086c
s[ 9]= g s+ 9= 0x40086d
s[10]= k s+10= 0x40086e
s[11]= l s+11= 0x40086f
s[12]= g s+12= 0x400870
s[13]= n s+13= 0x400871
s= 0x400864
p= 0x40086a
q= 0x400870
r= 0x400865
```

strspn()

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

int main(void) {

  char *s1 = "aaaa1111a22a3jj";
  char *s2 = "1234567a22a3jj";
  long   m1, m2;

  // s1 = "aaaa1111a22a3jj"
  printf("s1= %s \n", s1);

  m1= strspn(s1, "1234567890");  //      -> 0
  m2= strcspn(s1, "1234567890"); // aaaa --> 4

  printf("strspn(s1, \"1234567890\")  = %ld \n", m1);
  printf("strcspn(s1, \"1234567890\") = %ld \n", m2);

  // s2 = "1234567a22a3jj";
  printf("s2= %s \n", s2);

  m1= strspn(s2, "1234567890");  // 1234567 -> 7
  m2= strcspn(s2, "1234567890"); //         --> 0

  printf("strspn(s2, \"1234567890\")  = %ld \n", m1);
  printf("strcspn(s2, \"1234567890\") = %ld \n", m2);

}
```

C: complementary

char span

number span

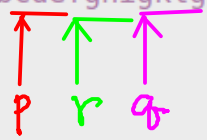
the length of the span consisting of numbers only

no numbers

```
s1= aaaa1111a22a3jj
strspn(s1, "1234567890")  = 0
strcspn(s1, "1234567890") = 4
s2= 1234567a22a3jj
strspn(s2, "1234567890")  = 7
strcspn(s2, "1234567890") = 0
```

strstr()

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

int main(void) {

  //         01234567890123 // 14 char's + 1 null
  char *s = "abcdefghigklgn";
  char *p;
  char *q;
  char *r;

  int len;
  int i;


  p= strstr(s, "cde");
  q= strstr(s, "gklg");
  r= strstr(s, "fghi");

  printf("s= %s \n", s);
  printf("p= %s \n", p);
  printf("q= %s \n", q);
  printf("r= %s \n", r);



  len = strlen(s);

  for (i=0; i<len; ++i)
    printf("s[%2d]= %c s+%2d= %p \n", i, s[i], i, s+i);

  printf("s= %p \n", s);
  printf("p= %p \n", p);
  printf("q= %p \n", q);
  printf("r= %p \n", r);


}
```



a ~~set of character~~

a substring

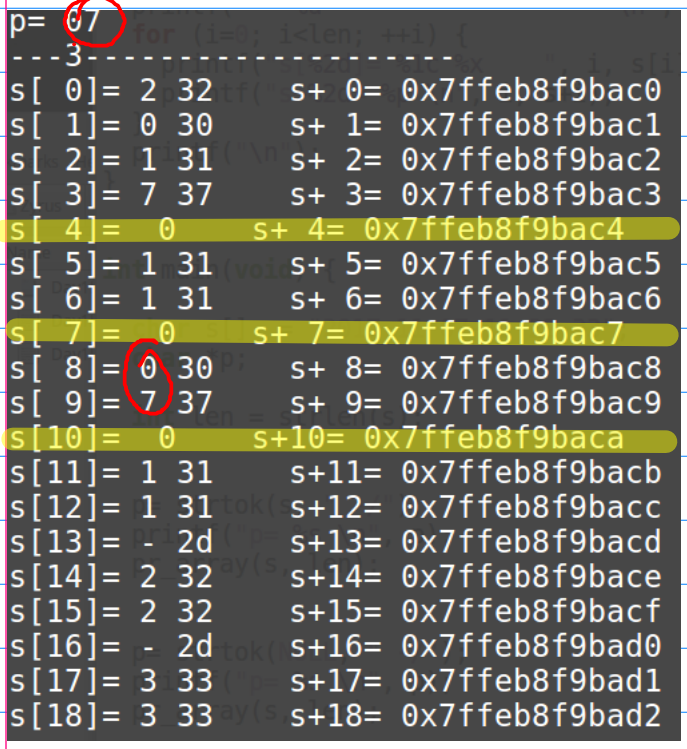
```
s= abcdefghigklgn
p= cdefghigklgn
q= gklgn
r= fghigklgn
s[ 0]= a s+ 0= 0x4007a4
s[ 1]= b s+ 1= 0x4007a5
s[ 2]= c s+ 2= 0x4007a6
s[ 3]= d s+ 3= 0x4007a7
s[ 4]= e s+ 4= 0x4007a8
s[ 5]= f s+ 5= 0x4007a9
s[ 6]= g s+ 6= 0x4007aa
s[ 7]= h s+ 7= 0x4007ab
s[ 8]= i s+ 8= 0x4007ac
s[ 9]= g s+ 9= 0x4007ad
s[10]= k s+10= 0x4007ae
s[11]= l s+11= 0x4007af
s[12]= g s+12= 0x4007b0
s[13]= n s+13= 0x4007b1
s= 0x4007a4
p= 0x4007a6
q= 0x4007ad
r= 0x4007a9
```

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

void pr_array(char *s, int len) {
  static int n = 1;
  int i;

  printf("---%d--------------------\n", n++);
  for (i=0; i<len; ++i) {
    printf("s[%2d]= %1c %x    ", i, s[i], s[i]);
    printf("s+%2d= %p \n", i, s+i);
  }
  printf("\n");
}


int main(void) {

  char s[]  = "2017-11-07-11-22-33";
  char *p;

  int len = strlen(s);


  p= strtok(s, " -/");
  printf("p= %s \n", p);
  pr_array(s, len);


  p= strtok(NULL, " -/");
  printf("p= %s \n", p);
  pr_array(s, len);

  p= strtok(NULL, " -/");
  printf("p= %s \n", p);
  pr_array(s, len);


}
```

```
p= 2017
---1--------------------
s[ 0]= 2 32    s+ 0= 0x7ffeb8f9bac0
s[ 1]= 0 30    s+ 1= 0x7ffeb8f9bac1
s[ 2]= 1 31    s+ 2= 0x7ffeb8f9bac2
s[ 3]= 7 37    s+ 3= 0x7ffeb8f9bac3
s[ 4]=  0      s+ 4= 0x7ffeb8f9bac4
s[ 5]= 1 31    s+ 5= 0x7ffeb8f9bac5
s[ 6]= 1 31    s+ 6= 0x7ffeb8f9bac6
s[ 7]= - 2d    s+ 7= 0x7ffeb8f9bac7
s[ 8]= 0 30    s+ 8= 0x7ffeb8f9bac8
s[ 9]= 7 37    s+ 9= 0x7ffeb8f9bac9
s[10]= - 2d    s+10= 0x7ffeb8f9baca
s[11]= 1 31    s+11= 0x7ffeb8f9bacb
s[12]= 1 31    s+12= 0x7ffeb8f9bacc
s[13]= - 2d    s+13= 0x7ffeb8f9bacd
s[14]= 2 32    s+14= 0x7ffeb8f9bace
s[15]= 2 32    s+15= 0x7ffeb8f9bacf
s[16]= - 2d    s+16= 0x7ffeb8f9bad0
s[17]= 3 33    s+17= 0x7ffeb8f9bad1
s[18]= 3 33    s+18= 0x7ffeb8f9bad2
```

'\0'

```
p= 11
---2--------------------
s[ 0]= 2 32    s+ 0= 0x7ffeb8f9bac0
s[ 1]= 0 30    s+ 1= 0x7ffeb8f9bac1
s[ 2]= 1 31    s+ 2= 0x7ffeb8f9bac2
s[ 3]= 7 37    s+ 3= 0x7ffeb8f9bac3
s[ 4]=  0     s+ 4= 0x7ffeb8f9bac4
s[ 5]= 1 31    s+ 5= 0x7ffeb8f9bac5
s[ 6]= 1 31    s+ 6= 0x7ffeb8f9bac6
s[ 7]=  0     s+ 7= 0x7ffeb8f9bac7
s[ 8]= 0 30    s+ 8= 0x7ffeb8f9bac8
s[ 9]= 7 37    s+ 9= 0x7ffeb8f9bac9
s[10]= - 2d    s+10= 0x7ffeb8f9baca
s[11]= 1 31    s+11= 0x7ffeb8f9bacb
s[12]= 1 31    s+12= 0x7ffeb8f9bacc
s[13]= - 2d    s+13= 0x7ffeb8f9bacd
s[14]= 2 32    s+14= 0x7ffeb8f9bace
s[15]= 2 32    s+15= 0x7ffeb8f9bacf
s[16]= - 2d    s+16= 0x7ffeb8f9bad0
s[17]= 3 33    s+17= 0x7ffeb8f9bad1
s[18]= 3 33    s+18= 0x7ffeb8f9bad2
```


p= 07
---3--------------------
s[0]= 2 32 s+ 0= 0x7ffeb8f9bac0
s[1]= 0 30 s+ 1= 0x7ffeb8f9bac1
s[2]= 1 31 s+ 2= 0x7ffeb8f9bac2
s[3]= 7 37 s+ 3= 0x7ffeb8f9bac3
s[4]= 0 s+ 4= 0x7ffeb8f9bac4
s[5]= 1 31 s+ 5= 0x7ffeb8f9bac5
s[6]= 1 31 s+ 6= 0x7ffeb8f9bac6
s[7]= 0 s+ 7= 0x7ffeb8f9bac7
s[8]= 0 30 s+ 8= 0x7ffeb8f9bac8
s[9]= 7 37 s+ 9= 0x7ffeb8f9bac9
s[10]= 0 s+10= 0x7ffeb8f9baca
s[11]= 1 31 s+11= 0x7ffeb8f9bacb
s[12]= 1 31 s+12= 0x7ffeb8f9bacc
s[13]= - 2d s+13= 0x7ffeb8f9bacd
s[14]= 2 32 s+14= 0x7ffeb8f9bace
s[15]= 2 32 s+15= 0x7ffeb8f9bacf
s[16]= - 2d s+16= 0x7ffeb8f9bad0
s[17]= 3 33 s+17= 0x7ffeb8f9bad1
s[18]= 3 33 s+18= 0x7ffeb8f9bad2