文字列処理

計算機とアルゴリズム I

第 14 回

7 月 19 日

文字列処理関数

C 言語では文字列の複写（コピー)、連結、比較等はすべて関数で行う。
その際、

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

が必要となる。

例

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int main(int argc, char *argv[]){
.................
```

文字列処理関数

strcpy(ss,tt)	文字列 ss に tt をコピー
strcat(ss,tt)	文字列 ss の後ろに tt を連結
strcmp(ss,tt)	文字列 ss と tt を比較 文字数によらず, 文字の大小関係で結果が決まる ss > tt なら 戻り値は正の数 ss < tt なら 戻り値は負の数 ss = tt なら戻り値は 0
strlen(ss)	文字列 ss の長さを返す
strncpy(ss,tt,n)	文字列 ss に tt の先頭から n 文字をコピー
strncat(ss,tt,n)	文字列 ss の後ろに tt の先頭から n 文字を連結
strncmp(ss,tt,n)	文字列 ss と tt の先頭から n 文字を比較

strcpy(ss,tt) について

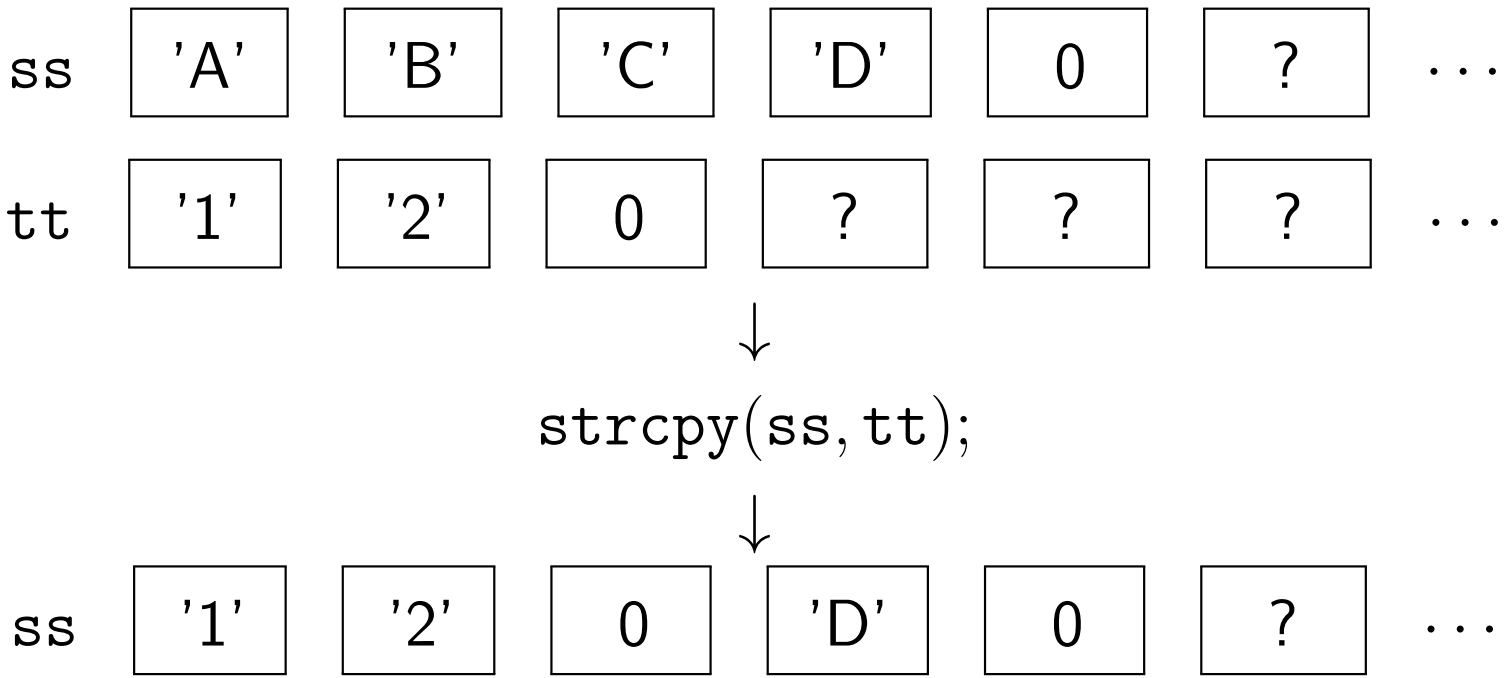
これは、文字列 (変数)ss に文字列 (変数)tt(の中身) を複写する関数である。

文字列 ss に"ABCD"が入っていて、文字列 tt に"12"が入っているとする。
ここで、

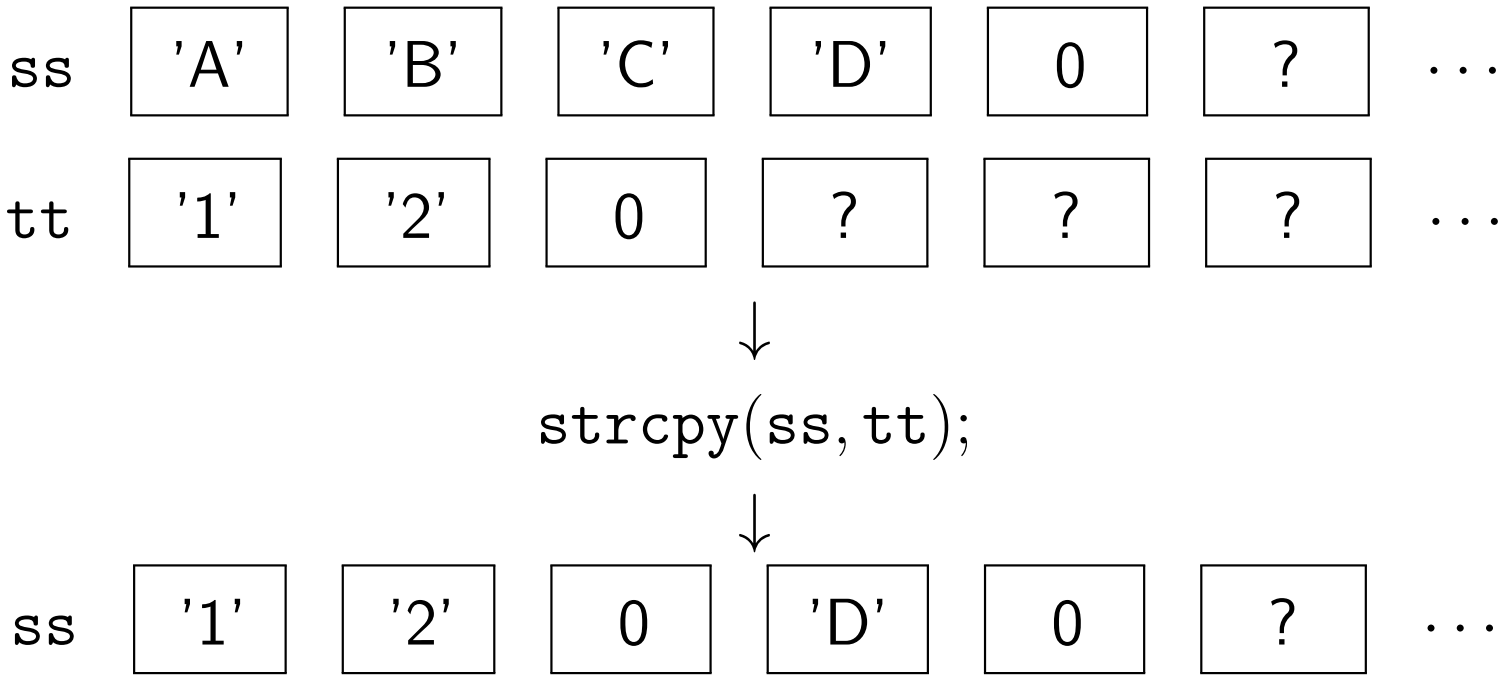
```
strcpy(ss,tt);
```

を行うと、文字列 ss は"12"になる。

strcpy(ss,tt) について



strcpy(ss,tt) について



♡ **注意!** このとき、文字列 tt をすべてコピーするのではなく、文字列 "12" (と終端) のみコピーしていることに注意する。

実際、printf("%c¥n",ss[3]); を実行すると、D が表示される。

文字列の入替え

数値 (a と b の値) の入替えは、

```
temp=a:
a=b:
b=temp:
```

のように行った。文字列 (a と b の内容) の入替えは、

```
strcpy(temp,a):
strcpy(a,b):
strcpy(b,temp):
```

のように行う。

strcat(ss,tt) について

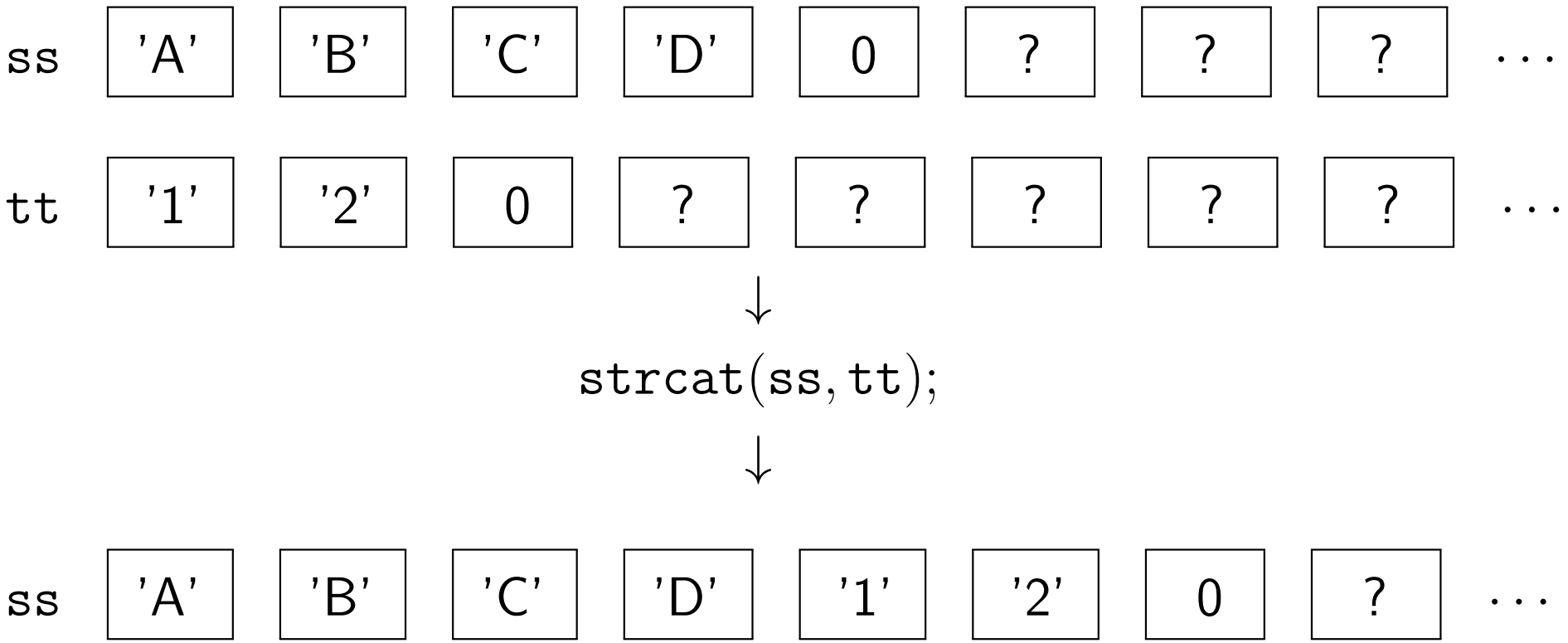
これは、文字列 (変数)ss の後ろに文字列 (変数)tt(の中身) を連結する関数である。

文字列 ss に"ABCD"が入っていて、文字列 tt に"12"が入っているとする。
ここで、

```
strcat(ss,tt);
```

を行うと、文字列 ss は"ABCD12"になる。

strcat(ss,tt) について



strncpy(ss,tt,n) と strncat(ss,tt,n)

strncpy(ss,tt,n) は、strcpy(ss,tt) と似たもので、

文字列 ss に文字列 tt の先頭から n 文字をコピー

する関数である。

同様に strncat(ss,tt,n) は、

文字列 ss の後ろに文字列 tt の先頭から n 文字を連結

する関数である。

いずれの場合も、n を tt の文字数以上にすると、strcpy(ss,tt) や strcat(ss,tt) と同じになる。

strncpy(ss,tt,n) と strncat(ss,tt,n)

文字列 ss が "ABCD" であり、文字列 tt が "12345" のとき、

```
strncpy(ss,tt,3);
```

を行うと、文字列 ss は "123D" になる。

strncpy(ss,tt,n) と strncat(ss,tt,n)

文字列 ss が "ABCD" であり、文字列 tt が "12345" のとき、

```
strncpy(ss,tt,3);
```

を行うと、文字列 ss は "123D" になる。

また、文字列 ss が "ABCD" であり、文字列 tt が "12345" のとき、

```
strncat(ss,tt,3);
```

を行うと、文字列 ss は "ABCD123" になる。

課題 M.1.

以下のプログラムを実行したとき、表示される文字列を答えよ。

課題 M.1.

```
#include<stdio.h>
#include<string.h>
int main(void){
  char ch1[20]="123456",ch2[20]="abcd",ch3[20];
  strncpy(ch1,ch2,3);
  printf("%s¥n",ch1);
  strncat(ch2,ch1,2);
  printf("%s¥n",ch2);
  strcpy(ch3,ch2);
  strcat(ch3,ch1);
  printf("%s¥n",ch3);
  return 0;
}
```

文字列処理関数

♡ **注意!**　文字列処理関数は "文字列処理" であるので、

```
strcpy(ss,"A");strcat("ABC","DEF");strncat("abc",tt,2,);
```

のようなことはできるが、

```
strcpy(ss,'A'); strcat("ABC",'D'); strncat('a',tt,2,);
```

のような (文字を文字列として扱う) ことはできない。

strlen(ss) について

これは、文字列 ss の長さを返す関数である。

文字列 ss に"ABCD"が入っているとする。ここで、

```
strlen(ss);
```

を行うと、4 という値が返ってくる。

strlen() の例

```
#include<stdio.h>
#include<string.h>
int main(void){
    char a[10]="abc",b[10]="defg";
    printf("¥"%s¥"の長さは%d です。¥n", a, strlen(a));
    strcat(a,b);
    printf("¥"%s¥"の長さは%d です。¥n", a, strlen(a));
    return 0;
}
```

実行結果

```
"abc"の長さは 3 です。
"abcdefg"の長さは 7 です。
```

strcmp(ss,tt) について

これは、文字列 ss と文字列 tt を比較し (それぞれの文字数によらず)、文字 (コード) の差の値を返す関数である。

まず、ss と tt が同じ文字列ならば、値は 0 となる。
ss と tt が異なる文字列なら、先頭から比べて最初に異なる文字同士を比較する。
その 2 文字 (例えば、k 番目の文字 ss[k] と tt[k]) に対して、値

$$ss[k] - tt[k]$$

を返す。
※ これは、それぞれの文字コードの値の差である。

strcmp(ss,tt) について

例 (strcmp の例)

```
 1 #include<stdio.h>
 2 #include<string.h>
 3 int main(void){
 4     printf("1) %d¥n", strcmp("ABC","ABC"));
 5     printf("2) %d¥n", strcmp("ABC","ABD"));
 6     printf("3) %d¥n", strcmp("ABC","AAAA"));
 7     printf("4) %d¥n", strcmp("ABC","AB"));
 8     printf("5) %d¥n", strcmp("AB","CC"));
 9     printf("6) %d¥n", strcmp("AB","ab"));
10     return 0;
11 }
```

strcmp(ss,tt) について

実行結果

1) 0
2) -1
3) 1
4) 67
5) -2
6) -32

※ 各文字の文字コードの値は

A	B	C	D	E	...	Z	a	b	c	d	e	...	z
65	66	67	68	69	...	90	97	98	99	100	101	...	122

である。(もちろん、記号や数字の場合も文字コードで扱う。)

課題 M.2.

例 5.1.5. や前回の例 5.1.5'. を参考に、大文字アルファベットのみの文字列 10 個を辞書式に並べ替えるプログラムを作成せよ。

文字列 10 個は、scanf で読み込んでも良いが、

```
char c[10][20]={"BC","AAA","AB","ABD","ABC",
                "BAC","CBA","BCB","BB","CA"};
```

のように定義しておいてよい。

実行結果

```
BC  ,AAA ,AB  ,ABD ,ABC ,BAC ,CBA ,BCB ,BB  ,CA  ,
AAA ,AB  ,ABC ,ABD ,BAC ,BB  ,BC  ,BCB ,CA  ,CBA ,
```

文字列処理関数

♠ 補足　strncpy, strncat には次のような使用方法もある。

文字列処理関数	
`strncpy(ss,tt+k,n);`	ss に tt の k 番から n 文字をコピー
`strncat(ss,tt+k,n);`	ss の後ろに tt の k 番から n 文字を結合

n 文字目からの文字の複製、結合

例

```
#include<stdio.h>
#include<string.h>
int main(void){
  char a[50]="abcdefg";
  char b[50]="ABCDEFG";
  printf("%s¥n",strncpy(a,b+2,3));
  printf("%s¥n",strncat(a,b+2,3));
  return 0;
}
```

実行結果

```
CDEdefg
CDEdefgCDE
```

本日の課題

・課題 M.1.
・課題 M.2.
・課題 M.3. (プリント 課題 5.3.5.)
・課題 M.4. (プリント 課題 5.3.7.)

提出方法はメールで yoshiyuki-mori@ous.ac.jp まで送る。

注意！6 月からアドレス変更。

件名は学生番号と課題 M. と書き、

例) S24M999 課題 M.

ファイルを添付し送ること。

締め切りは 7 月 27 日 23:59
次回はまとめの課題 II (締め切りは 7 月 28 日講義終了時)