プログラミング基礎演習A

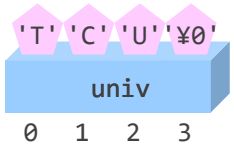
文字列

文字型配列と文字列

◆ 文字型配列

```
char univ[4];

univ[0] = 'T';
univ[1] = 'C';
univ[2] = 'U';
```



配列の大きさは(文字数+1)にする

◆ 文字型配列を文字列として扱うには…

末尾にヌル文字(0 or '¥0')を入れる!

```
univ[3] = 0;
  または
univ[3] = '¥0';
```

文字列にするといろいろな関数が使えるよ!

文字列の初期化

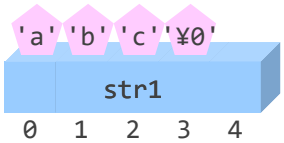
```
char 配列名[要素数] = "文字列"
```

省略可

文字列は" "で括る

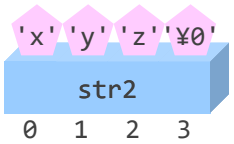
```
char str1[5] = "abc";
```

箱は5個




```
char str2[] = "xyz";
```

省略すると



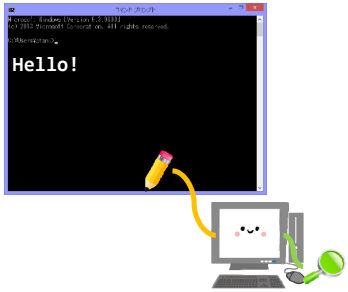
箱は3+1=4個

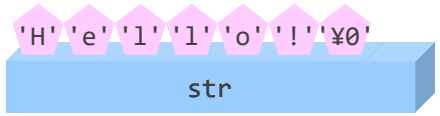
printf関数における文字列の書式指定

◆ 変換指定子は%s

```
char str[8] = "Hello!";

printf("%s¥n", str);
```





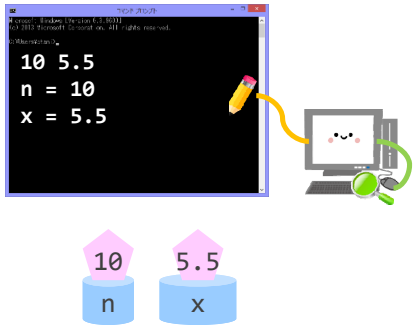
もう一度scanf関数

scanf("フォーマット文字列", 変数アドレス, 変数アドレス, …)

- キーボードから値を読み込み変数に代入する関数
- 改行, 空白文字までをひとかたまりのデータとみなす

```
int n;
double x;

scanf("%d%lf", &n, &x);
printf("n = %d¥n", n);
printf("x = %f¥n", x);
```

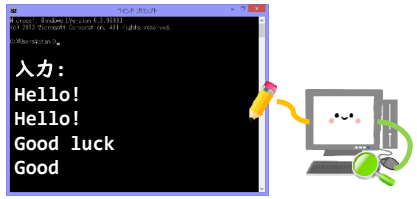


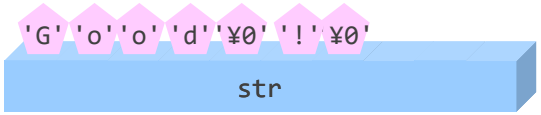
scanf関数における文字列の書式指定

- 変換指定子は%s

```
char str[10];

printf("入力:¥n");
scanf("%s", str);
printf("%s¥n", str);
scanf("%s", str);
printf("%s¥n", str);
```





スペース,
改行の前までが
1つの文字列

最後に
ヌル文字が
追加される

文字列での禁止事項

- 初期化以外の=による代入

```
char str1[5];
char str2[5];

str1 = "abc";
str2 = str1;
```

- ==による比較

```
if(str1 == str2) {
    ...
```

配列名は配列の先頭のアドレス
(位置・ポインタ)を表す

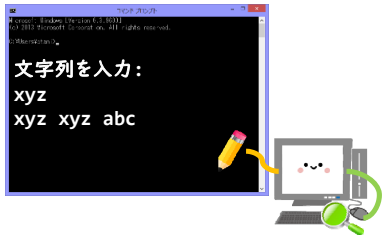
strcpy関数による文字列の複製

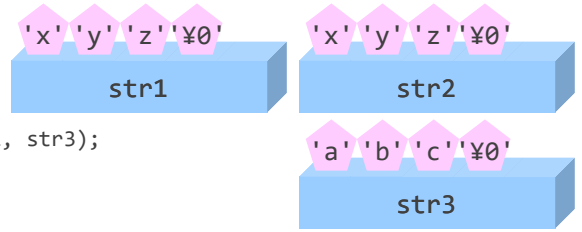
strcpy(複製文字列を入れる変数, 複製元文字列)

- 文字列を複製する関数
- ヘッダファイルstring.hが必要

```
char str1[5];
char str2[5];
char str3[5];

printf("文字列を入力:¥n");
scanf("%s", str1);
strcpy(str2, str1);
strcpy(str3, "abc");
printf("%s %s %s¥n", str1, str2, str3);
```





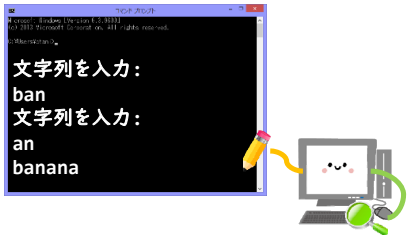
strcat関数による文字列の連結

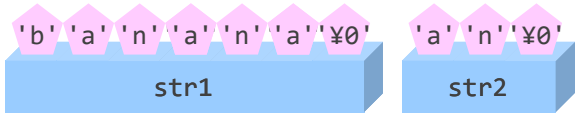
strcat(連結先文字列, 連結文字列)

- ある文字列に別の文字列を連結する関数
- ヘッダファイルstring.hが必要

```
char str1[7];
char str2[3];

printf("文字列を入力:¥n");
scanf("%s", str1);
printf("文字列を入力:¥n");
scanf("%s", str2);
strcat(str1, str2);
strcat(str1, "a");
printf("%s¥n", str1);
```





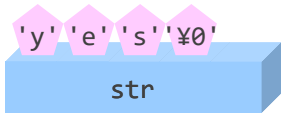
strcmp関数による文字列の比較

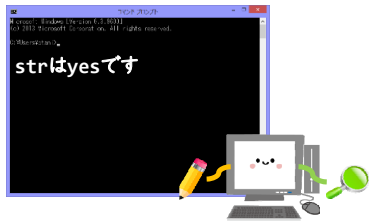
strcmp(文字列1, 文字列2)

- 2つの文字列を比較する関数
- 戻り値 ： 2つの文字列が等しいとき0,
  等しくないとき0以外の値
  - 辞書順に並べたとき, 文字列1が前に来るときは負の値,
    文字列2が前に来るときは正の値
- ヘッダファイルstring.hが必要

```
char str[5] = "yes";

if(strcmp(str, "yes") == 0) {
    printf("strはyesです¥n");
}
```





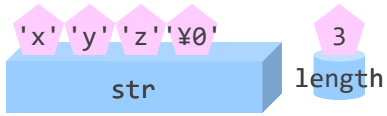
strlen関数による文字列の長さの取得

strlen(文字列)

- 文字列の長さ(ヌル文字を含めないバイト数)を求める関数
- 戻り値 ： 文字列の長さ
- ヘッダファイルstring.hが必要

```
char str[5];
int length;

printf("文字列を入力:¥n");
scanf("%s", str);
length = strlen(str);
printf("長さは%d¥n", length);
```



puts関数による文字列の出力

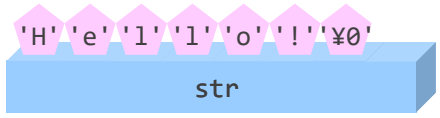
発展

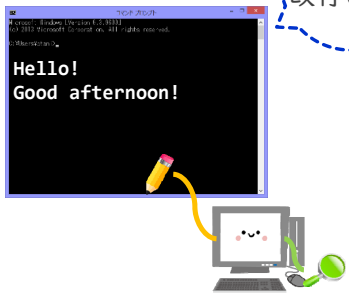
puts(文字列)

- 文字列を出力する関数
- ヘッダファイルstdio.hが必要

```
char str[8] = "Hello!";

puts(str);
puts("Good afternoon!");
```





自動的に改行される

gets関数による文字列の入力

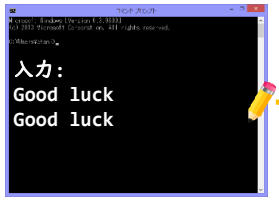
発展

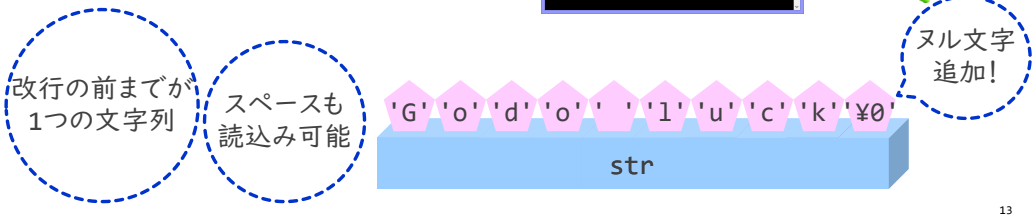
gets(文字列変数)

- キーボードから文字列を読み込み変数に代入する関数
- ヘッダファイルstdio.hが必要

```
char str[10];

printf("入力:¥n");
gets(str);
printf("%s¥n", str);
```





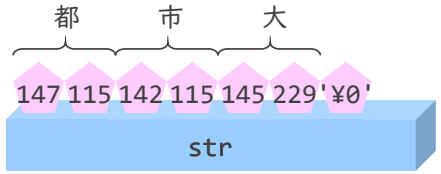
全角文字

発展

全角文字(2バイト)は長さ2の文字列として扱う

→ 半角文字(1バイト)2つ分

```
char str[8] = "都市大";
```




```
都 : 0x9373 = 147 115
市 : 0x8E73 = 142 115
大 : 0x91E5 = 145 229
```


演習

文字列(最大80文字), 置換開始位置m, 置換文字数nを入力すると, 入力した文字列のm文字目からn文字分を「*」に置換した文字列が表示されるプログラムを作成せよ. ただし, 入力された文字列の長さよりもmやm+n-1が大きい場合には, エラーメッセージが表示されるようにすること.

プログラム名はe12とすること.

<実行例1>
文字列:TokyoCityUniversity
置換開始位置:3
置換文字数:5
置換後の文字列:To*****tyUniversity

<実行例2>
文字列:TokyoCityUniversity
置換開始位置:25
置換文字数:3
入力された文字列は25文字未満です

<実行例3>
文字列:TokyoCityUniversity
置換開始位置:5
置換文字数:20
5文字目から20文字分を置換することができません