計算機とアルゴリズム I 第6回
コンソール入出力

森 義之

岡山理科大学

1 文字の入出力

C 言語で標準入力 (キーボード) から文字を入力するには

`getchar()`, `gets()`, `scanf()`

を使い、標準出力 (ディスプレイ) に文字を出力するには

`putchar()`, `puts()`, `printf()`

を使う。1 文字のみの入出力には `getchar()`, `putchar()` を使う。

1 文字の入出力

C 言語で標準入力 (キーボード) から文字を入力するには

`getchar()`, `gets()`, `scanf()`

を使い、標準出力 (ディスプレイ) に文字を出力するには

`putchar()`, `puts()`, `printf()`

を使う。1 文字のみの入出力には `getchar()`, `putchar()` を使う。

今後、`scanf()`,`printf()` を良く使うので、その他は省略する。
(詳しくは、教科書 96-105p)

文字列 (配列) の変数宣言と扱い

複数文字からなる 1 行の文字列を扱う場合は、変数として **配列** を使う。

文字列 (配列) の変数宣言と扱い

複数文字からなる 1 行の文字列を扱う場合は、変数として **配列** を使う。

例えば、配列を用いた変数宣言の例として

```
char ch[100];
```

や、

```
char city[50]="Okayama";
```

である。

文字列 (配列) の変数宣言と扱い

この

```
char ch[100];
```

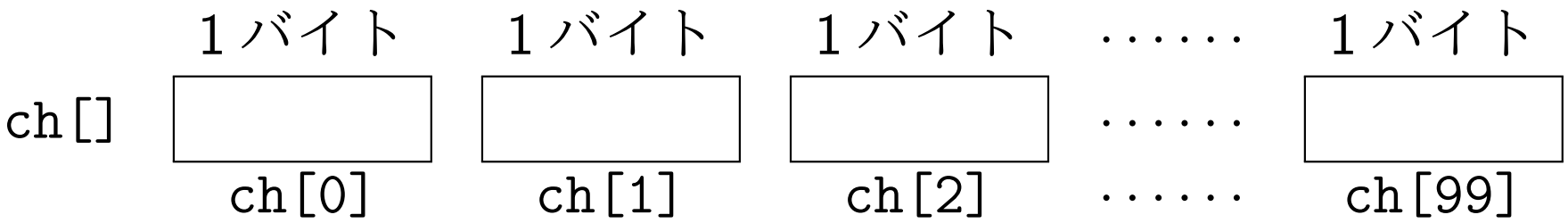
は、1 バイト幅 (char 型) を 100 個用意するという意味である。
すなわち、

文字列 (配列) の変数宣言と扱い

この

```
char ch[100];
```

は、1 バイト幅 (char 型) を 100 個用意するという意味である。
すなわち、



のように、箱が用意される。

ただし、箱の番号は 0 から 99 となることに注意する。

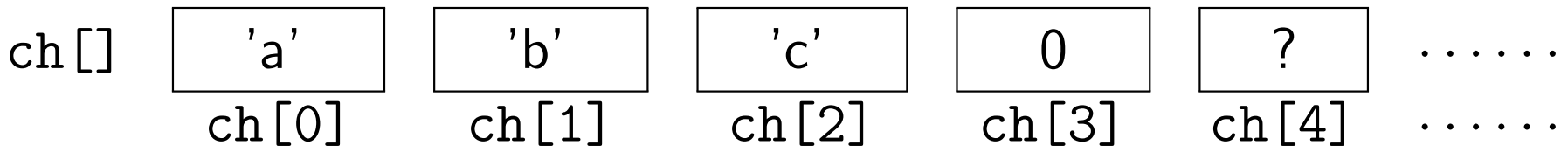
文字列 (配列) の扱い

このとき、各箱に 1 文字が入ることになる。

たとえば、

```
char ch[100]="abc";
```

とすれば、



となる。

課題 D.0.

以下のプログラムを入力し、実行結果を確認せよ。

課題 D.0.

```
#include<stdio.h>
  int main(void){
    char ch[100]="abcde";
    printf("文字列%s について、",ch);
    printf("1 文字目は%c、",ch[0]);
    printf("3 文字目は%c です。¥n",ch[2]);
    return 0;
  }
```

実行結果

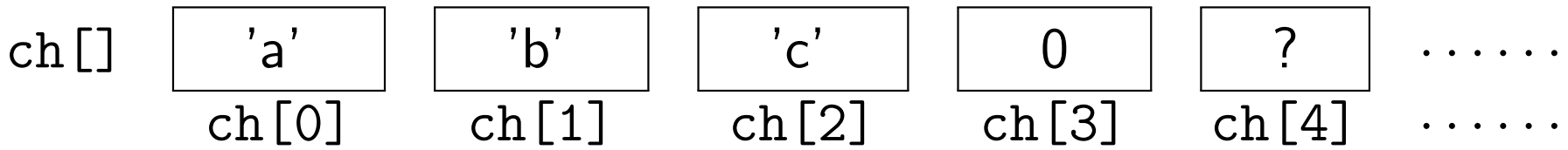
文字列 abcde について、1 文字目は a、3 文字目は c です。

文字列 (配列) の扱い

また、

```
char ch[100]="abc";
```

としたとき



であるが、文字列の終わり (終端) に 0 が代入されている。
(ch[3] が'0' でないことに注意)
ちなみに、この 0 は任意に入力することが出来る。

文字列 (配列) の扱い

例 (文字列の終端の扱い)

```
 1 #include<stdio.h>
 2 void main(void){
 3     char ch[100]="abcdefg";
 4
 5     printf("%s¥n",ch);
 6     ch[3]=0;
 7     printf("%s¥n",ch);
 8     ch[3]='D';
 9     printf("%s¥n",ch);
10 }
```

文字列 (配列) の扱い

例 (文字列の終端の扱い)

```
 1 #include<stdio.h>
 2 void main(void){
 3     char ch[100]="abcdefg";
 4
 5     printf("%s¥n",ch);
 6     ch[3]=0;
 7     printf("%s¥n",ch);
 8     ch[3]='D';
 9     printf("%s¥n",ch);
10 }
```

実行結果

```
abcdefg
abc
abcDefg
```

複数文字 (1 行) の入出力の例

まずは例。 複数文字 (文字列) の入力は、scanf を用いる。

例 (scanf, printf を使い 1 行文字列を入力, 出力する例)

```
1 #include<stdio.h>
2 int main(void){
3     char name[100];
4     scanf("%s",name);
5     printf("%s",name);
6     return 0;
7 }
```

実行結果

```
abc    ← キーボードからの入力が表示されている (4 行目)
abc    ← 5 行目の printf による出力
```

課題 D.1.

以下の.... の部分に scanf 文を正しく入力し、実行結果を確認せよ。

課題 D.1.

```
#include<stdio.h>
  int main(void){
    char ch[100];
    printf("3 文字以上の文字列を入力してください:");
    ....................................
    printf("入力された文字列の 1 文字目は%c、",ch[0]);
    printf("3 文字目は%c です。¥n",ch[2]);
    return 0;
  }
```

実行結果

```
3 文字以上の文字列を入力してください:okayama
入力された文字列の 1 文字目は o、3 文字目は a です。
```

複数文字(1行)の入出力

♡ **注意!** 変換指定%c と%s の違いに注意する。

```
char ch[100]="abcdefg";
printf("%c ¥n",ch); // NG
printf("%s ¥n",ch); // OK
printf("%c ¥n",ch[1]); // OK
printf("%s ¥n",ch[1]); // NG
```

%c は 1 文字を表示し、%s は文字列を表示する。

scanf() を用いてデータを入力する場合

int 型変数 n に、数値を入力する場合

```
scanf("%d", &n);
```

double 型変数 n に、数値を入力する場合

```
scanf("%lf", &n);
```

char 型（1 文字) 変数 n に、文字を入力する場合

```
scanf("%c", &n);
```

char 型 (文字列) 変数 n に、文字列を入力する場合

```
scanf("%s", n);
```

※ 浮動小数の場合、%f でなく%lf であることに注意。

文字列, 文字の入力

課題 D.2. 点線部分に適切なものを入れ完成させよ。

```
 1 #include<stdio.h>
 2    int main(void){
 3    char ini, name[100];
 4    printf("名前のイニシャルを入力してください:");
 5    ..................................  //ini への入力
 6    printf("名字を入力してください:");
 7    ..................................  //name への入力
 8    printf("こんにちは ..... さん ¥n",.........);
 9    return 0;
10 }
```

実行結果

```
名前のイニシャルを入力してください:Y
名字を入力してください:Mori
こんにちは Y.Mori さん
```

数値の入力

課題 D.3. 点線部分に適切なものを入れ完成させよ。

```
 1 #include<stdio.h>
 2   int main(void){
 3   int i;
 4   double d;
 5   printf("整数を入力してください:");
 6   ..................................... //i への入力
 7   printf("浮動小数を入力してください:");
 8   ..................................... //d への入力
 9   printf("i+d は.... です。¥n",.......);
10   return 0;
11 }
```

実行結果

```
整数を入力してください:28
浮動小数を入力してください:3.14159
i+d は 31.141590 です。
```

scanf() の注意点

♡ **注意!** `scanf()` 関数は `printf()` 関数と似ているが、

```
scanf("n の値を入力してください %d",&n);
```

のような使い方はできない。

scanf() の注意点

♡ **注意!** scanf() 関数は printf() 関数と似ているが、

```
scanf("n の値を入力してください %d",&n);
```

のような使い方はできない。希望の表示方法は

```
printf("n の値を入力してください ");
scanf("%d",&n);
```

とする。(入力関数と出力関数は別もの。)

scanf() の注意点

♡ **注意!** scanf() 関数は printf() 関数と似ているが、

```
scanf("n の値を入力してください %d",&n);
```

のような使い方はできない。希望の表示方法は

```
printf("n の値を入力してください ");
scanf("%d",&n);
```

とする。(入力関数と出力関数は別もの。)

♠ **補足** scanf() 関数は重大なにエラーに対する対処がないため、テストプログラムや入門書を除いてはあまり使われない。しかし、解り易く便利であり、プログラマと使用者が正しく使用すれば問題は少ないため、この講義では使用している。

(代替 fgets, sscanf)

scanf() の注意点

scanf() 関数は、2 回以上連続で使用する場合、正しく入力しても最後の改行文字はバッファの中に残るため、意図しない動作をするおそれがある。

scanf() の注意点

例 (入力エラー)

```
 1 #include <stdio.h>
 2 int main(void){
 3     int a, b;
 4     char ch;
 5     printf("数値 a=");
 6     scanf("%d",&a);
 7     printf("文字=");
 8     scanf("%c",&ch);
 9     printf("数値 b=");
10     scanf("%d",&b);
11     return 0
12 }
```

scanf() の注意点

例 (入力エラー)

```
 1 #include <stdio.h>
 2 int main(void){
 3     int a, b;
 4     char ch;
 5     printf("数値 a=");
 6     scanf("%d",&a);
 7     printf("文字=");
 8     scanf("%c",&ch);
 9     printf("数値 b=");
10     scanf("%d",&b);
11     return 0
12 }
```

実行結果

```
数値 a=12345
文字=数値 b=67890
```

scanf() の注意点

5, 6 行目の表示, 入力は正しく実行されるが、その後の 7, 8 行目が正しく実行されない。
これは、6 行目の scanf() の際に入力したエンターキー (¥n) がバッファに残り、ch に代入されてしまっている。このような動作を防ぐためには %c の前に空白を 1 つ入れて、

scanf() の注意点

5, 6 行目の表示, 入力は正しく実行されるが、その後の 7, 8 行目が正しく実行されない。
これは、6 行目の scanf() の際に入力したエンターキー (¥n) がバッファに残り、ch に代入されてしまっている。このような動作を防ぐためには %c の前に空白を 1 つ入れて、

```
8 scanf(" %c",&ch);
```

とする。

scanf() の注意点

例 (【改】入力エラー)

```
 1 #include <stdio.h>
 2 int main(void){
 3     int a, b;
 4     char ch;
 5     printf("数値 a=");
 6     scanf("%d",&a);
 7     printf("文字=");
 8     scanf(" %c",&ch);
 9     printf("数値 b=");
10     scanf("%d",&b);
11     return 0;
12 }
```

実行結果

```
数値 a=12345
文字=k
数値 b=67890
```

本日の課題

・課題 D.1.

・課題 D.2.

・課題 D.3.

・課題 D.4. (プリント ページ 課題 3.3.6.)

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

注意！アドレス変更。

件名は学生番号と課題 D. と書き、

例) S24M999 課題 D.

ファイルを添付し送ること。

締め切りは 6 月 1 日 23:59。