Cプログラミング演習1(再)　7

講義では、Cプログラミングの基本を学び
演習では、実践的なプログラミングを通して学ぶ

今までの復習

プログラムで最低限必要なもの

- 入力（キーボードから、ファイルから）
- 出力（画面へ、ファイルへ）
- 条件分岐：条件の成立・不成立により、異なる動作をする
- 繰り返し: 一定の回数の繰返し、条件成立の間の繰返し
- 関数の定義、関数の呼び出し

Cではそれ以外に、ポインタ、データ構造（配列や自作の構造）、ライブラリの利用、（C++ではクラス）が必要

アルゴリズム

アルゴリズムとは

料理で言えばレシピ

要するに、処理の段取り

どのような処理をどのような順で行うか

アルゴリズムによって処理効率に大きな違いがあることも

文字列を対象とした「数え上げ」

lang.sist.chukyo-u.ac.jp/Classes/C/animals.txt
は次のような内容のファイルである

どんな動物が何回現れているだろうか？

いぬ
ねこ
ねこ
さる
いぬ
ねこ
ねこ
さる
ねこ
わに
ねこ
いぬ
ねこ
…

事前知識：文字と文字列

教科書「よくわかるC言語」第6章(p.63-75)

- 文字 --- 英数字や記号の1文字 ’a’, ’B’, ’1’, ’¥n’ など
  文字を記憶する変数の型は　char　--- character(文字)
- 文字列 --- 文字の並び、文字の配列
  例:　”abc”, ”123”, ”tab¥t end¥n”
  文字列の最後には必ず ’¥0’ (ナル文字) が入る（が、表示はされない）
  ーこれが文字列の最後を表すキマリ
  文字列を記憶する変数の型は　char　string[10];　などとする

クイズ: ”a” は文字でしょうか、文字列でしょうか？
　また ’字’ は文字でしょうか、文字列でしょうか？

答: 文字列
　どちらでもない

事前知識:文字列

- 文字列 --- 文字の並び、文字の配列

ここで質問

”abc”, ”123”, ”tab¥t end¥n” はどんな型でしょうか？

型の例: 1 は int

1.0 は float

‘A’ は char

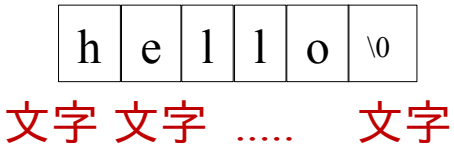
ここで「文字列」と答えた人、気持ちはわかりますが、そういう名前の「型」はありません え！

事前知識：文字列

- 文字列 --- 文字の並び、文字の配列

そう、答はここに書いてありました。だから、

例えば　”hello”　は、次の文字の配列を表しています:



だから、char str[ ] = ”hello” ;

とすると、変数strには、上の配列が記憶されたメモリの場所がセットされます

事前知識:文字列

- 文字列 --- 文字の並び、文字の配列

このことから、以下が「同じ意味」であることはわかりますね？

```
char  str[ ] = “abc”;
char  str[ ] = { ‘a’, ‘b’, ‘c’, ‘¥0’ };
```

文字の配列を指すポインタ

char str[ ] = ”hello”;
とすると、変数strには、上の配列が記憶されたメモリの場所がセットされます

と書きました。言い換えれば、str (だけではなく「配列」全部)
は、文字(とか整数)の値を持っているのではなく、
文字の配列の場所（アドレス、ポインタ）が値です。
ここで
場所(ポインタ)の値から、その場所にあるデータを読み出す方法があります。それが ＊ です。

検証

右のプログラム(stringTest1.c)を試してみましょう。

まず、str を文字列として、初期値を入れています

　その値を printf で文字列として表示させています。

次に、strが指している場所の文字、次の場所(次の場所は+1で得られます)の文字、... を表示させています

```
#include <stdio.h>

int main(void) {
  char str[ ] = "abc";
  printf("%s¥n", str);
  printf("%x %x %x¥n",
      str[0],str[1],str[2]);
  printf("%c %c %c¥n",
      str[0],str[1],str[2]);
  printf("%c-%c-%c-%d¥n",
     *str,*(str+1),*(str+2),*(str+3));
  return 0;
}
```

文字を指すポインタ

このように、* はその変数の値がどのようなものを表しているか(char, int, floatなど)を表す記号であることがわかりました。
（この記号は、すでにファイルを扱うときに出ていました）

この知識を用いて、以下を考えてみましょう:

```
char  str[ ] = “abc”;
char  *try = “abc”;
```

クイズ: 変数 try がどのようなものか、説明してみてください。
また、これは str の値と同じでしょうか、違うでしょうか？

検証

右のプログラム(stringTest3.c)を試してみましょう。

まず、str を文字列とし、tryにも初期値を入れています

tryの値を printf で文字列として表示させています。

次に、tryが指している場所の文字、次の場所(次の場所は+1で得られます)の文字、... を表示させています

最後にstr と tryが等しいかを調べています

```
#include <stdio.h>

int main(void) {
  char str[ ] = "abc";
  char *try = "abc";

  printf("%s¥n", try);
  printf(“%c-%c-%c-%d¥n”,
        *try,*(try+1),*(try+2),*(try+3));
  printf("str == try?  ans = %d¥n",
       str == try);
  return 0;
}
```

文字の配列とポインタの関係

右のプログラム(stringTest2.c)を試してみましょう。
“abc” という文字の配列において、それぞれの要素（例えば str[1])がどのような値なのか、またそのアドレスがどうなっているかを示すプログラムになっています。
　このことから、
(1)str[1] と *(str+1)が同じものであること、
(2) 文字の配列の要素が, メモリ上では1番地ずつずれていること
がわかります。

```
#include <stdio.h>

int main(void) {
  char str[ ] = "abc";
  printf("%s\n", str);
  printf("%x %x %x\n", str[0],str[1],str[2]);
  printf("%c-%c-%c-%d\n",
        *str,*(str+1),*(str+2),*(str+3));
  printf("------------\n");
  printf("%x %x %x\n",
        &str[0], &str[1], &str[2]);
  printf("%x %x %x\n", str, str+1, str+2);
  return 0;
}
```

文字列の配列

右のプログラム(stringTest4.c)を試してみましょう。

ここでは、文字列の配列 str を定義しています(この場合 5 という文字の長さ指定が必要です---書かずにコンパイルしたらどうなるか、やってみましょう）

なお、走らせる前に、どのような表示がなされるか、推測してみましょう

```
#include <stdio.h>

int main(void) {

  char str[ ][5] = {"ab","xyz","漢字"};

  printf("%x %x %x\n", str[0],str[1],str[2]);
  printf("%s %s %s\n", str[0],str[1],str[2]);
  printf("%s-%s-%s\n", *str,*(str+1),*(str+2));
  return 0;
}
```

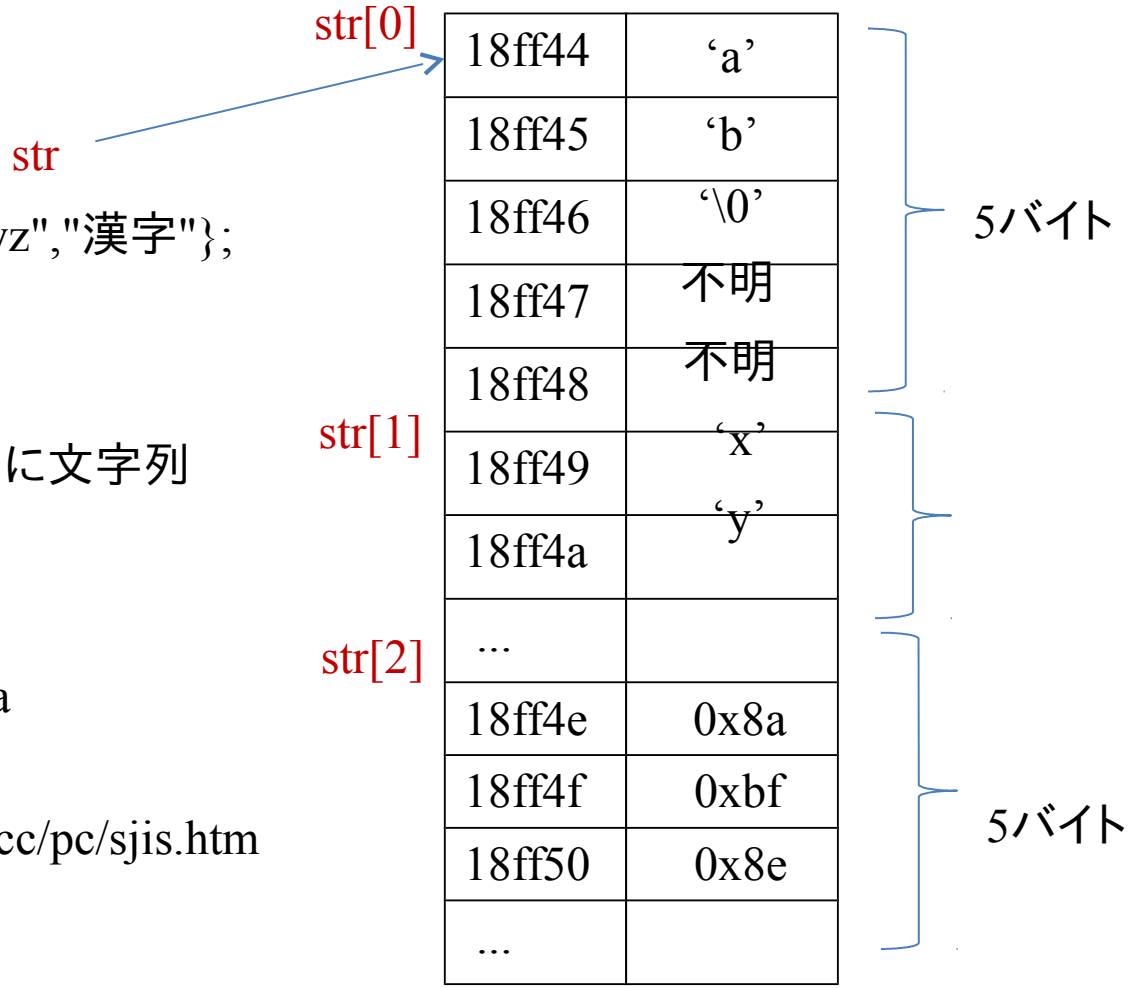
解説

char str[ ][5] = {"ab","xyz","漢字"};

によって、
メモリ上に「5バイト」×3
の場所が確保され、そこに文字列
が記憶される

なお、「漢」は　0x8abf
「字」は　0x8e9a
Shift-JISで表現される

参考: http://qpon.quu.cc/pc/sjis.htm



文字列に関する関数

教科書のp.71 :

string.h ヘッダファイルにて定義されている

strcpy(char s1[ ], char s2[ ])

strcat(char s1[ ], char s2[ ])

int strcmp (char s1[ ], char s2[ ])　　後でこれを使う

int strlen(char s[ ])

文字列を対象とした「数え上げ」

lang.sist.chukyo-u.ac.jp/Classes/C/animals.txt

の中にどんな動物が何回現れているかを数えるプログラムをつくろう

ただし、どんな動物が現れるか分からないままではちょっと難しいので(そのうち出来るようになるはずだが)、ここでは

「いぬ、ねこ、うさぎ、くま、さる、たぬき、きつね、ぱんだ、ねずみ、わに」だけが出てくる可能性があるものとしよう

システムの設計

概略、次のようなアルゴリズムを考える

1. ファイル(animals.txt)から一行ずつ読み込み、文字列を要素とする配列(仮に変数名をstrとする)にその文字列を記憶する
2. 候補の動物名の一個ずつに対し、
    1. strの要素それぞれに対して、一致するかどうか調べる。
    2. 一致した個数を、動物名とともに表示する

1. ファイル(animals.txt)から一行ずつ読み込み、文字列を要素とする配列(仮に変数名をstrとする)にその文字列を記憶する

これを具体的にどうするかを考えよう。

必要な物は、

(1)文字列を要素とする配列

文字列の長さをLEN、ファイルの行数をLINESとすると
char str[LINES][LEN];　　// チェック!なぜこの順番？

(2)ファイル(animals.txt)から一行ずつ読み込んで、配列で記憶する方法

これは今までにもやってきたこと。。。

1. ファイル(animals.txt)から一行ずつ読み込み、文字列を要素とする配列(仮に変数名をstrとする)にその文字列を記憶する

これを具体的にどうするかを考えよう。

(2)ファイル(animals.txt)から一行ずつ読み込んで、配列で記憶する

文字列の長さをLEN、ファイルの行数をLINESとすると
char str[LINES][LEN];　　// チェック!なぜこの順番？

いままでやってきたことです

2. 候補の動物名の一個ずつに対し、

strの要素それぞれに対して、一致するかどうか調べる。

一致した個数を、動物名とともに表示する

次に、これを具体的にどうするかを考えよう。

必要な物は、

(1)候補の動物名を記憶する方法

(2) 文字列同士を比較して、一致しているかどうか、調べる方法

(1)候補の動物名を記憶する

教科書に従って、次のようにしてもよい

候補は:「いぬ、ねこ、うさぎ、くま、さる、たぬき、きつね、ぱんだ、ねずみ、わに」
だから、要素を文字列とする配列を用意すれば良い。
その配列（変数）の名前を Animals とすると、(LENの値を最初に与えておく)

文字列の配列は

char 変数名[ ][LEN] = { 初期値 }; で宣言

配列　二重配列　初期値となる文字列の並び

だから。。。

(1)候補の動物名を記憶する(別解)

候補は:「いぬ、ねこ、うさぎ、くま、さる、たぬき、きつね、ぱんだ、ねずみ、わに」

だから、要素を文字列とする配列を用意すれば良い。

その配列(変数)の名前を Animals とすると、

文字列の配列は char* 変数名[ ]={ 初期値 }; で宣言

要素は文字列　配列　初期値となる文字列の並び

だから。。。

ここまでできたら、
　ちゃんと動くか
テストしてみよう

テスト1．配列 Animals

右のプログラムを完成させ
animalsCheck.c
と名前をつけ、
実行してみよう

どんな結果が表示されれば
成功といえるか、考えてから
実行しよう

```
#include <stdio.h>
// 候補となる動物の個数
#define NUM 10
// 動物名の最大長さ
#define  LEN  8

int main(void) {
   // 配列 Animals の宣言
   // 他に必要な変数があれば、その宣言

   // Animalsの値をprintfを用いて確認
   for(i=0;
       print
   }
   return 0;
}
```

書けるよね？

テスト2.
　　ファイルからの読み込み

右のプログラムを完成させ
animalsInput.c
と名前をつけ、
実行してみよう

どんな結果が表示されれば
成功といえるか、考えてから
実行しよう

```
#include <stdio.h>
// ファイルの行数の最大値を仮に100とする
#define  LINES 100
// 動物名の最大長さ
#define  LEN  8
// 読み込み対象のファイル名
#define InFile "animals.txt“
int main(void) {
   // 配列strなど、必要な変数の宣言
   // ファイルから読み込んで配列strに記憶
   // その時に読み込んだ行数を変数nに記憶
   // ファイルを閉じる
   // strの値をprintfを用いて確認
   for(i=0;
        print
   }
   return 0;
}
```

書けるよね？

(2) 文字列同士を比較して、一致しているかどうか、調べる

stringh. ヘッダファイルで定義された

strcmp 関数を使う

これがどんな引数をとり、どういう条件でどんな値を返すかを確認する

以上ができたら、プログラム
を完成させせよう(countAnimals.c)

1. ファイル(animals.txt)から一行ずつ読み込み、文字列を要素とする配列(仮に変数名をstrとする)にその文字列を記憶する
2. 候補の動物名の一個ずつに対し、
   ① strの要素それぞれに対して、一致するかどうか調べる。
   ②一致した個数を、動物名とともに表示する

```
#include <stdio.h>
#define NUM 10
#define LINES 100
#define LEN 8
int main(void) {
  // 配列 Animals と str 他必要な変数の宣言
  // 1. ファイルから読み込んだ文字列を
  //       配列strに記憶
  // ファイルを閉じる
  // 2.候補の動物名の一個ずつに対し(繰り返し)
  for(i=0; i<NUM; i++) {
      // ① 一致した個数を記憶する変数mに0代入
      //  ② str の要素それぞれに対して動物名が

      //        一致すれば、m++ とする
      // ③ m>0なら、動物名とmの値を表示
  }
  return 0;
}
```

プログラムがかけたら…

コンパイル（ビルド）して
animals.txt のデータで試してみよう

不具合があれば修正し、再度テストする