

Cプログラミング入門

(基幹5クラス)

第12回 文字と文字列

本日の講義・演習項目

- ▶ 文字と文字列
 - ▶ char型
 - ▶ ASCIIコード
 - ▶ 文字列
 - ▶ 文字列と関数

char型変数

- ▶ **文字**を記憶することのできる変数
 - ▶ 半角文字を**1文字**記憶できる

```
char x, y;
```

... char型変数の宣言

```
x = 'a';
```

... 変数xに文字**a**を代入

```
y = 'A';
```

... 変数yに文字**A**を代入

- ▶ 文字の代入には ‘ ’ が必要
- ▶ 標準入出力

```
printf("%c", x);
```

... 端末に文字を出力

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

... 端末から文字を入力

ASCIIコード

- 内部的には、文字は**数字**に置き換えて扱われる
 - 対応関係は**ASCIIコード**で規定される

ASCIIコード（一部）

文字	数字	文字	数字	文字	数字	文字	数字	文字	数字	文字	数字	文字	数字
A	65	I	73	Q	81	Y	89	g	103	o	111	w	119
B	66	J	74	R	82	Z	90	h	104	p	112	x	120
C	67	K	75	S	83	a	97	i	105	q	113	y	121
D	68	L	76	T	84	b	98	j	106	r	114	z	122
E	69	M	77	U	85	c	99	k	107	s	115	{	123
F	70	N	78	V	86	d	100	l	108	t	116		124
G	71	O	79	W	87	e	101	m	109	u	117	}	125
H	72	P	80	X	88	f	102	n	110	v	118	~	126

char型変数／ASCIIコード

▶ 文字と対応する数字を表示するプログラム

```
#include <stdio.h>
int main(void){
    char x = 'a', y = 'A';
    printf("x = %c, code: %d\n", x, x);
    printf("y = %c, code: %d\n", y, y);
    return 0;
}
```

▶ 実行結果



```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc char0.c -o char0
[w536074@W63C061A 12]$ ./char0
x = a, code: 97
y = A, code: 65
[w536074@W63C061A 12]$
```

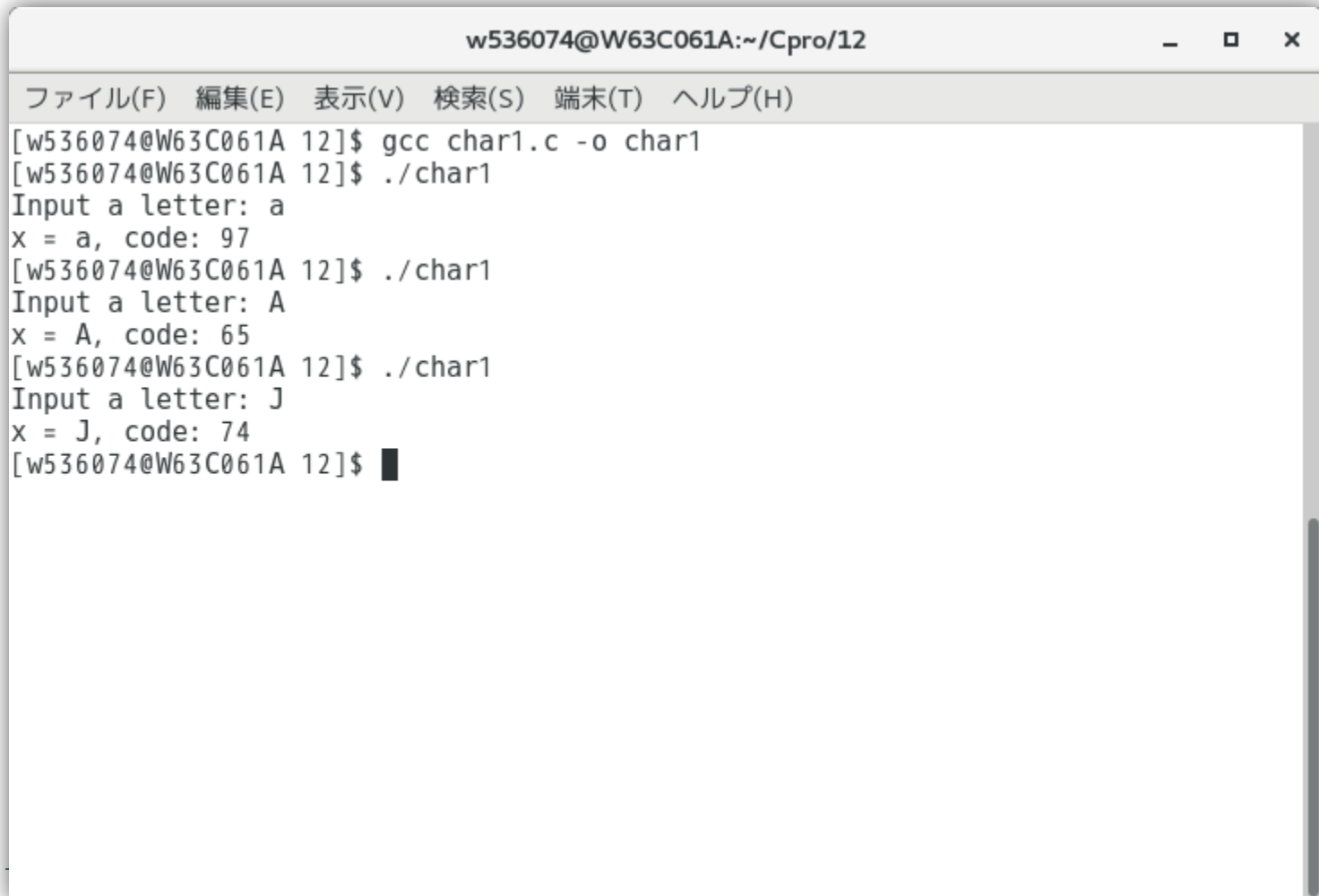
演習①

- ▶ 端末から半角文字を入力し、文字と対応する数字を表示するプログラム char1.c を作成せよ。

```
#include <stdio.h>
int main(void){
    char x;
    printf("Input a letter: ");
    scanf("%c", &x);
    printf("x = %c, code: %d\n", x, x);
    return 0;
}
```

演習①

▶ 実行例



```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc char1.c -o char1
[w536074@W63C061A 12]$ ./char1
Input a letter: a
x = a, code: 97
[w536074@W63C061A 12]$ ./char1
Input a letter: A
x = A, code: 65
[w536074@W63C061A 12]$ ./char1
Input a letter: J
x = J, code: 74
[w536074@W63C061A 12]$
```

演習②

- ▶ 端末からアルファベットの大文字を入力し，小文字に変換して表示するプログラム char2.c を作成せよ。
- ▶ ただし，大文字アルファベット以外が入力された場合を考慮する必要はないものとする。

```
#include <stdio.h>
int main(void){
    char x, y, offset = 'a' - 'A';
    printf("Input a capital letter: ");
    scanf();

    y = ;

    printf("%c -> %c\n", x, y);
    return 0;
}
```

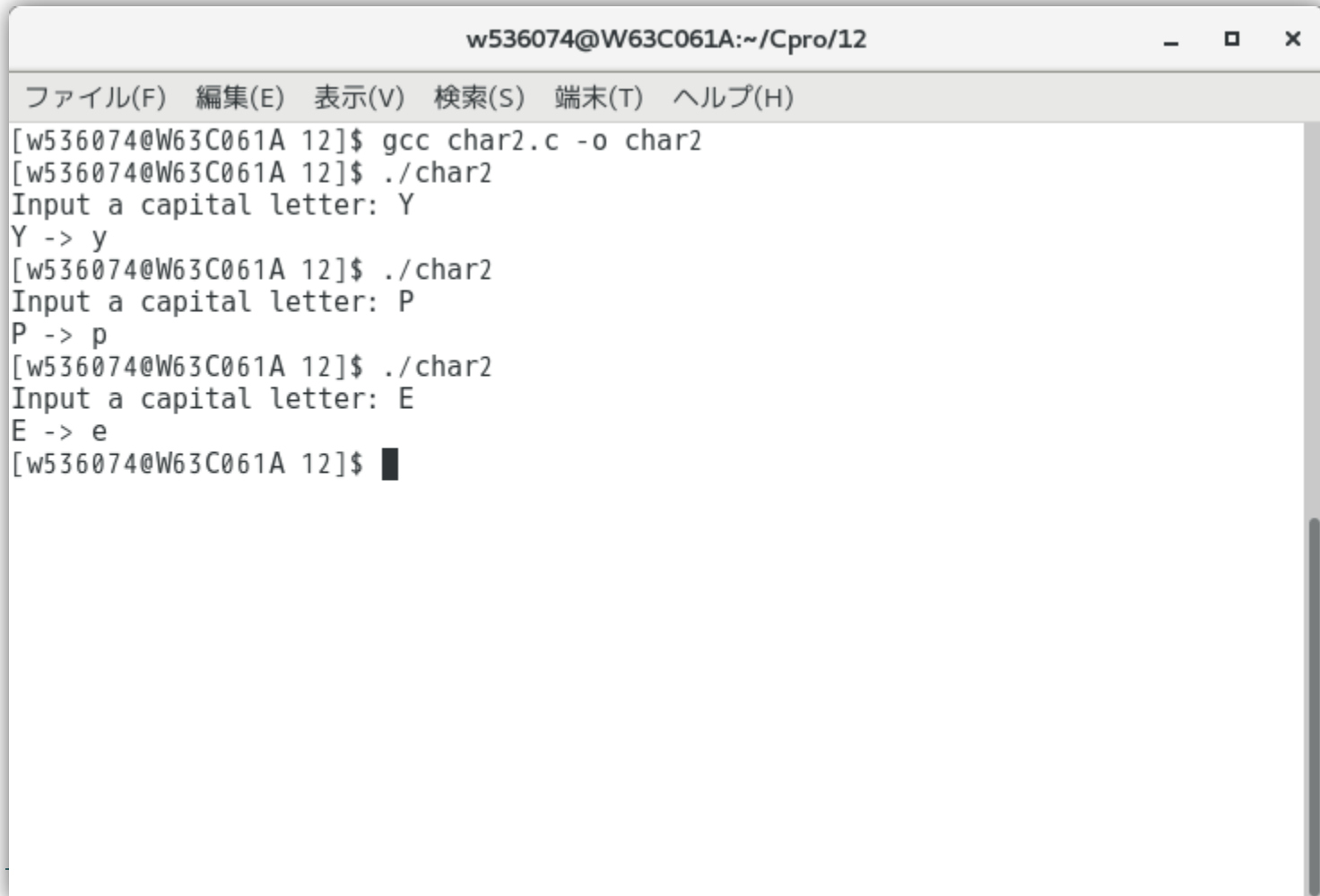
← char型は内部的には
数字として扱う

... 演習①と同様

← ASCIIコードにおける
大文字と小文字の関係は？

演習②

▶ 実行例

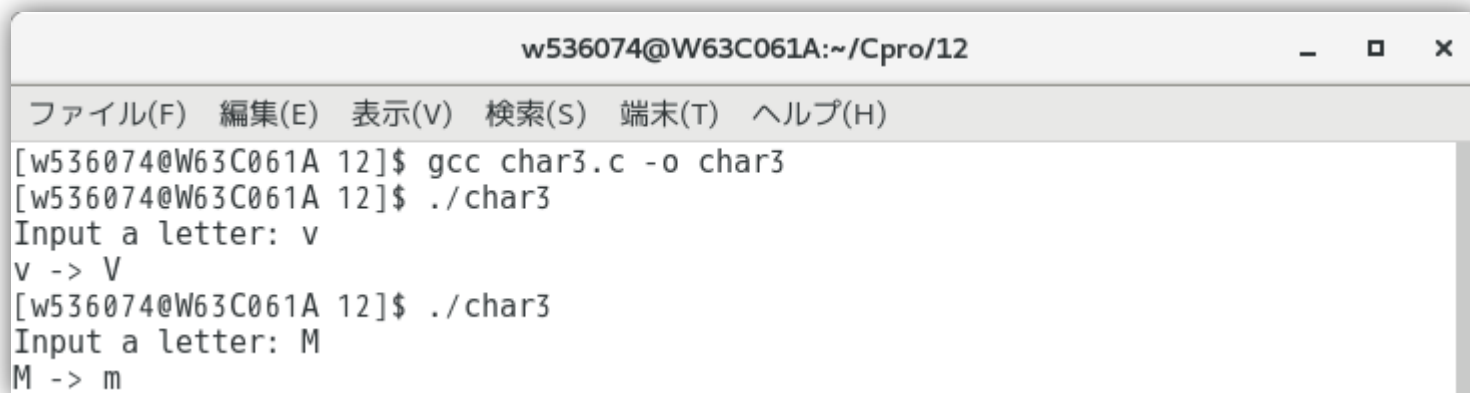


```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc char2.c -o char2
[w536074@W63C061A 12]$ ./char2
Input a capital letter: Y
Y -> y
[w536074@W63C061A 12]$ ./char2
Input a capital letter: P
P -> p
[w536074@W63C061A 12]$ ./char2
Input a capital letter: E
E -> e
[w536074@W63C061A 12]$
```

演習③（余力がある人向け）

- ▶ 端末からアルファベットを入力し，大文字であれば小文字に変換し，小文字であれば大文字に変換して表示するプログラム `char3.c` を作成せよ。
- ▶ ただし，アルファベット以外が入力された場合を考慮する必要はないものとする。

▶ 実行例



```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc char3.c -o char3
[w536074@W63C061A 12]$ ./char3
Input a letter: v
v -> V
[w536074@W63C061A 12]$ ./char3
Input a letter: M
M -> m
```

文字列

- ▶ char型変数は半角文字を1文字だけ記憶
- ▶ **文字列**を扱いたい場合 → char型の**配列**を使用

```
char x[256];
```

... char型配列の宣言

x[0]	x[1]	x[2]	x[3]	x[4]	x[5]	x[6]	x[7]	x[8]	x[9]	
H	e	l	l	o	!	¥0	a	b	c	...

- ▶ 文字列を扱う際のポイント

- ▶ 十分な配列サイズを確保
- ▶ 文字列の終わりを表す**終端文字**‘¥0’

Hello!
を表す文字列

文字列の初期化と代入

▶ 初期化

```
char x[256] = "Waseda University";
```

- ▶ 文字列は “ ” で表す
- ▶ 宣言時のみ可

▶ 標準入出力

```
printf("%s", x); ... 端末に文字列を出力  
scanf("%s", x); ... 端末から文字列を入力
```


- ▶ 配列名xはアドレスを表す → **&は不要**

文字列

▶ 自分の名前を表示するプログラム

```
#include <stdio.h>
int main(void){
    char x[256] = "Kazushi Kawamura";
    printf("My name is %s.\n", x);
    return 0;
}
```

▶ 実行結果

A terminal window with a title bar "w536074@W63C061A:~/Cpro/12" and standard window controls. The menu bar includes "ファイル(F)", "編集(E)", "表示(V)", "検索(S)", "端末(T)", and "ヘルプ(H)". The terminal shows the following commands and output:

```
[w536074@W63C061A 12]$ gcc string0.c -o string0
[w536074@W63C061A 12]$ ./string0
My name is Kazushi Kawamura.
[w536074@W63C061A 12]$
```

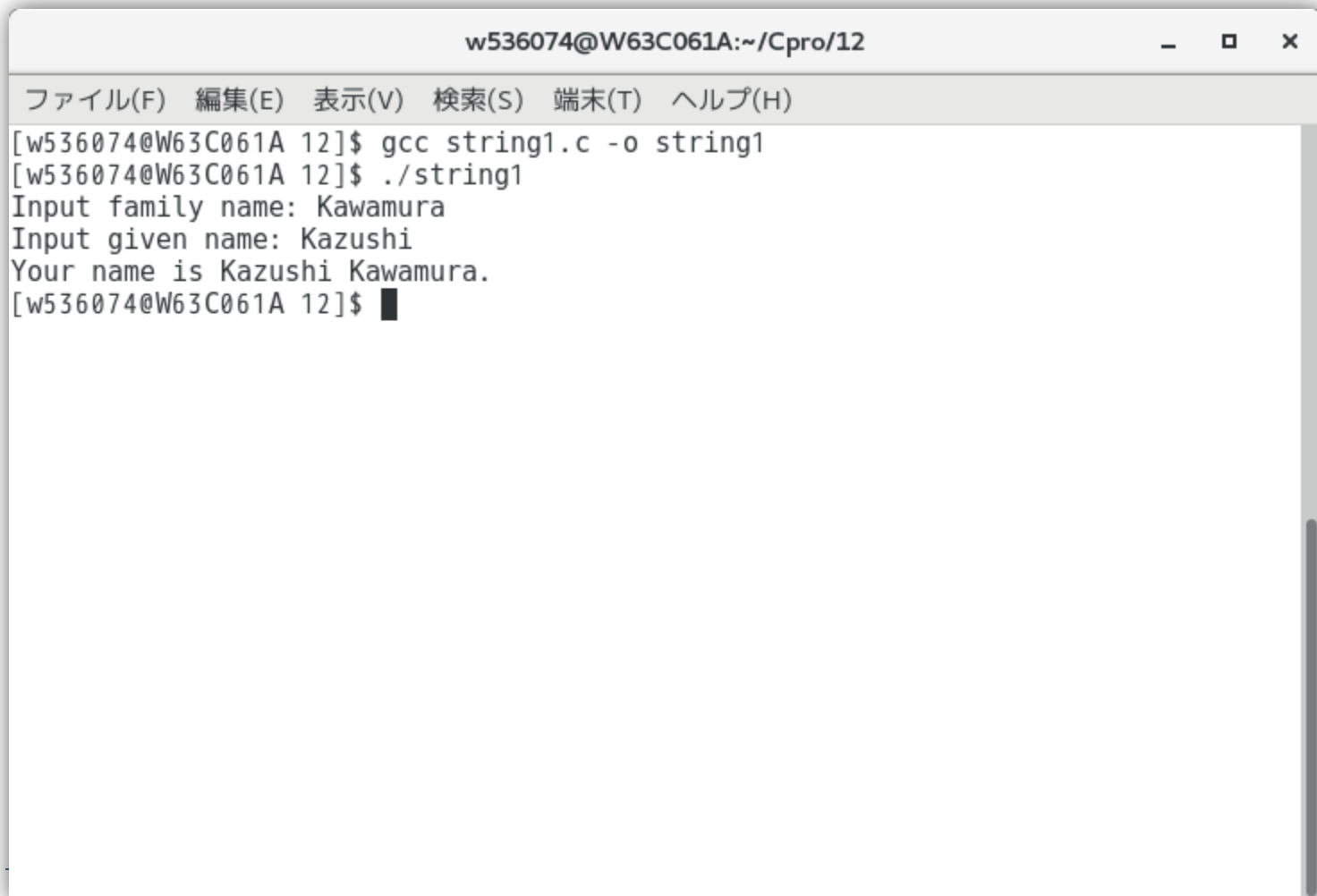
演習④

- ▶ 端末から姓（Family name）と名（Given name）を入力し，名前を表示するプログラム string1.c を作成せよ。

```
#include <stdio.h>
#define SIZE 256
int main(void){
    char f[SIZE], g[SIZE];
    printf("Input family name: ");
    scanf("%s", f);
    printf("Input given name: ");
    scanf("%s", g);
    printf("Your name is %s %s.\n", g, f);
    return 0;
}
```

演習④

▶ 実行例

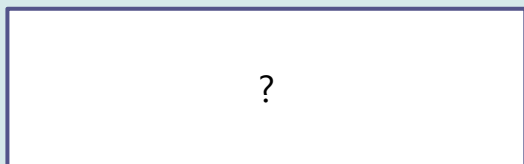


```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc string1.c -o string1
[w536074@W63C061A 12]$ ./string1
Input family name: Kawamura
Input given name: Kazushi
Your name is Kazushi Kawamura.
[w536074@W63C061A 12]$
```

演習⑤

- ▶ 端末から入力した文字列の長さをカウントし，表示するプログラム string2.c を作成せよ。
- ▶ ただし，strlen関数を使わずに作成すること

```
#include <stdio.h>
#define SIZE 256
int main(void){
    char x[SIZE];
    printf("Input x: ");
    scanf("%s", x);
```



```
    printf("Length: %d\n",  );
    return 0;
```

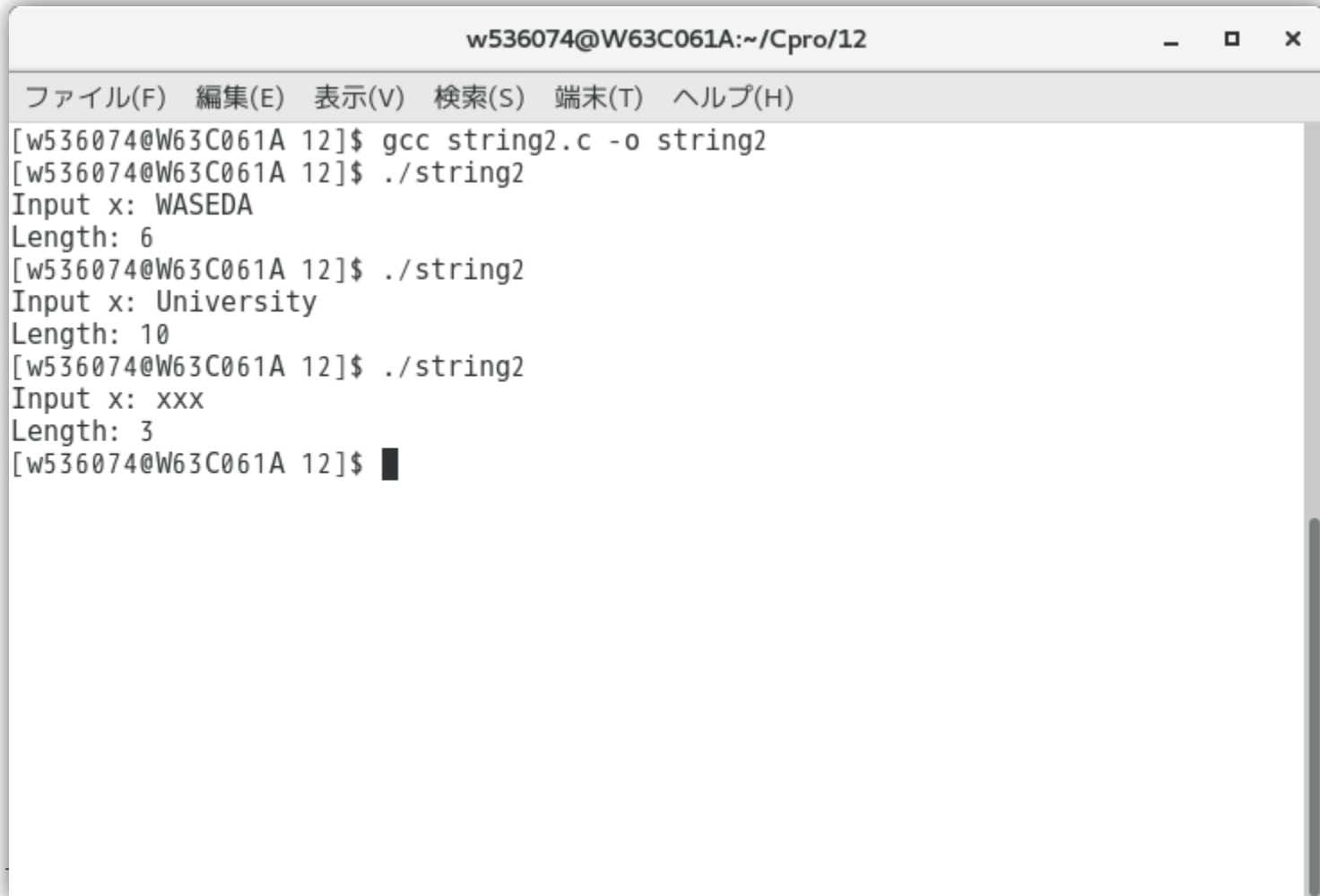
```
}
```

終端文字‘\0’に
注目する



演習⑤

▶ 実行例



```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc string2.c -o string2
[w536074@W63C061A 12]$ ./string2
Input x: WASEDA
Length: 6
[w536074@W63C061A 12]$ ./string2
Input x: University
Length: 10
[w536074@W63C061A 12]$ ./string2
Input x: xxx
Length: 3
[w536074@W63C061A 12]$
```

文字列と関数

▶ プログラム例：文字列をコピーする関数

```
#include <stdio.h>
#define SIZE 256


void s_copy(char *x, char *y){
    int i;
    for(i = 0; i < SIZE ; i++){
        y[i] = x[i];
    }
    return;
}

int main(void){
    char x[SIZE], y[SIZE];
    printf("Input: "); scanf("%s", x);

    s_copy(x, y);

    printf("x: %s\n", x);
    printf("y: %s\n", y);
    return 0;
}
```

文字配列x, yの
先頭アドレスを渡す



演習⑥

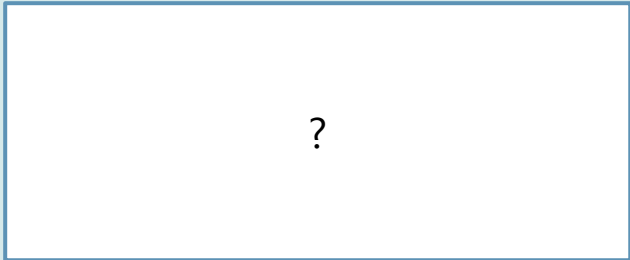
- ▶ 端末から入力した文字列を逆順にコピーして表示するプログラム string3.c を作成せよ。
- ▶ 実行例



```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc string3.c -o string3
[w536074@W63C061A 12]$ ./string3
Input x: waseda
x: waseda
y: adesaw
[w536074@W63C061A 12]$ ./string3
Input x: abcdefghi
x: abcdefghi
y: ihgfedcba
[w536074@W63C061A 12]$
```

演習⑥

```
#include <stdio.h>
#define SIZE 256

void s_reverse(char *x, char *y){
    
}

int main(void){
    char x[SIZE], y[SIZE];
    printf("Input x: "); scanf("%s", x);

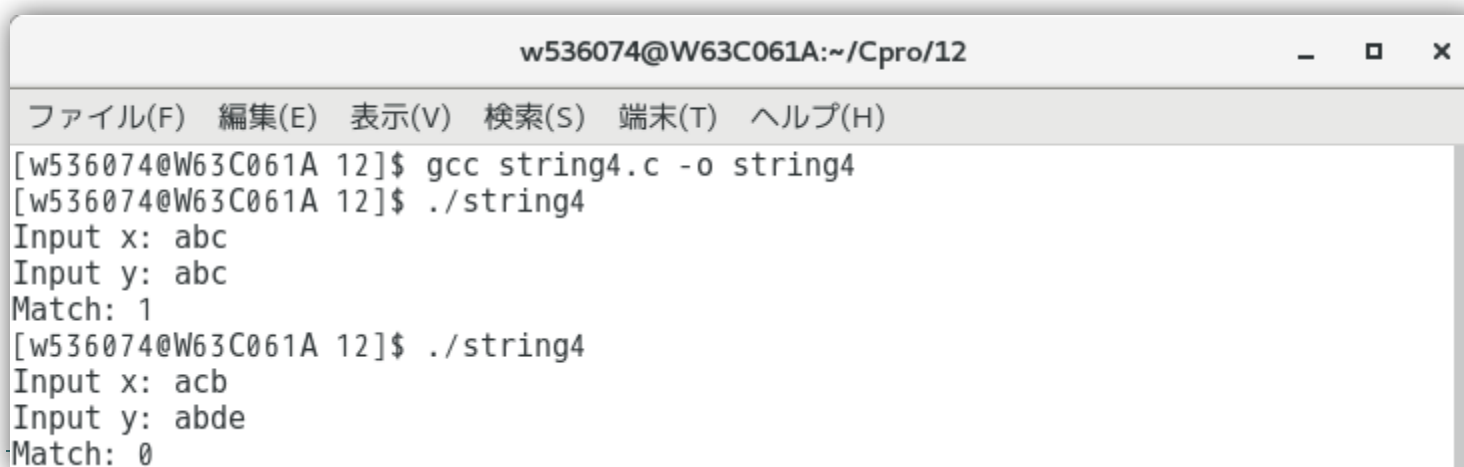
    s_reverse(x, y);

    printf("x: %s\n", x);
    printf("y: %s\n", y);
    return 0;
}
```

演習⑦

- ▶ 端末から入力した二つの文字列が一致しているかを判定するプログラム `string4.c` を作成せよ。
 - ▶ ただし, `strcmp`関数を使わずに作成すること
 - ▶ 一致している場合は1を出力し, 一致していない場合は0を標準出力する。

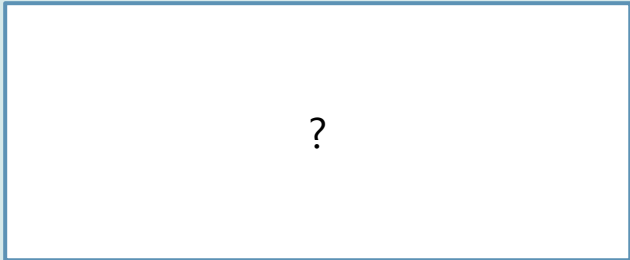
▶ 実行例



```
w536074@W63C061A:~/Cpro/12
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C061A 12]$ gcc string4.c -o string4
[w536074@W63C061A 12]$ ./string4
Input x: abc
Input y: abc
Match: 1
[w536074@W63C061A 12]$ ./string4
Input x: acb
Input y: abde
Match: 0
```

演習⑦

```
#include <stdio.h>
#define SIZE 256

int s_match(char *x, char *y){
    
}

int main(void){
    char x[SIZE], y[SIZE];
    printf("Input x: "); scanf("%s", x);
    printf("Input y: "); scanf("%s", y);

    int result;
    result = s_match(x, y);

    printf("Match: %d\n", result);
    return 0;
}
```

キーワード，次回の講義

- ▶ 本日のキーワード：
- ▶ 次回は7/12
- ▶ 次回講義までに予習ビデオ「第13回 ファイル入出力」を視聴し，各自プログラミング実習
- ▶ 前回講義で課題②を出題済み（×切：7/8）