

Cプログラミング入門

(基幹5クラス)

第14回 構造体

本日の講義・演習項目

- ▶ 授業内演習の解説 ... 第 1 3 回講義の解答スライドに記載
- ▶ 構造体

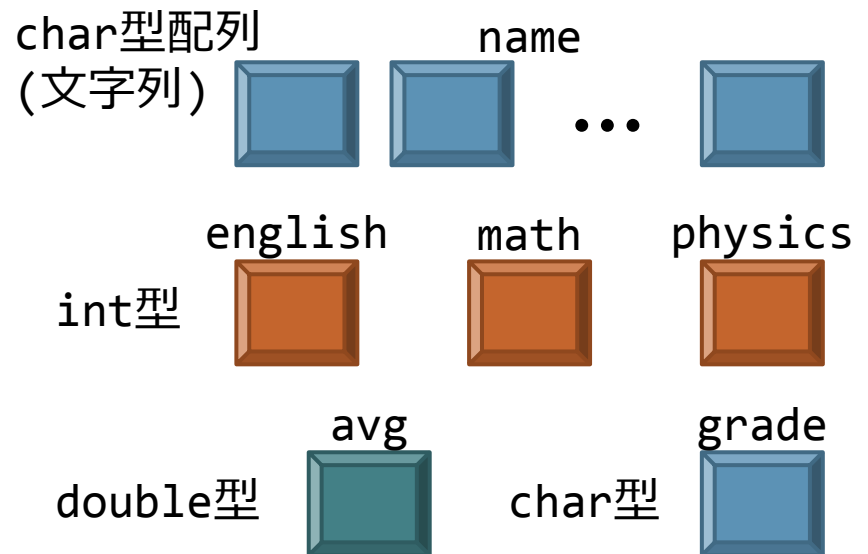
構造体

- ▶ 複数の型を組み合わせて作る**新たなデータ構造**
 - ▶ プログラム内で**自由に定義可能**

- ▶ 構造体の型の定義 6 個のメンバを持つデータ型

```
struct student{  
    char name[20];  
    int english;  
    int math;  
    int physics;  
    double avg;  
    char grade;  
};
```

struct student 型



構造体

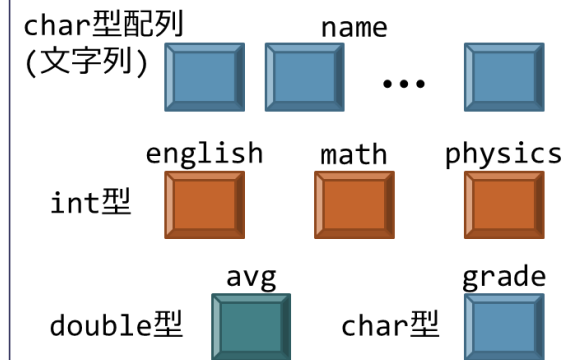
▶ 構造体変数の宣言

```
struct student{  
    char name[20];  
    int english;  
    int math;  
    int physics;  
    double avg;  
    char grade;  
};
```

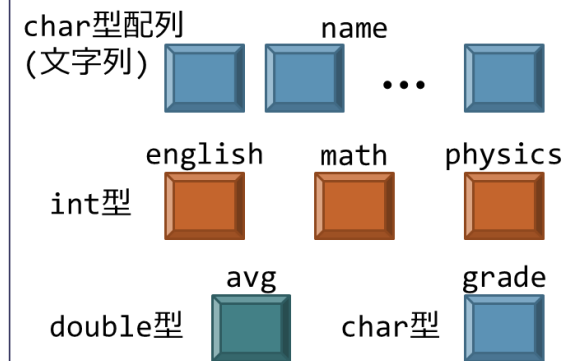
struct student 型
変数a, bの宣言

```
int main(void){  
    struct student a, b;  
    ...  
}
```

変数 a



変数 b



構造体

▶ 構造体変数の初期化と代入

```
struct student{  
    char name[20];  
    int english;  
    int math;  
    int physics;  
    double avg;  
    char grade;  
};
```

各メンバの初期化が先頭から順に行われ、
値が与えられないメンバは0で初期化される

```
int main(void){  
    struct student a = {"Alice", 82, 72, 58};  
    struct student b = {"Bob"};  
    b.english = 71;  
    b.math = 92;  
    b.physics = 98;  
    ...  
}
```

} 初期化

} 代入

各メンバの参照には、演算子 (.) を使う

演習①

- ▶ 学生の成績データを扱う以下のプログラム struct1.c を作成し、実行せよ。

```
#include <stdio.h>


struct student{
    char name[20];
    int english, math, physics;
    double avg;
    char grade;
};

int main(void){
    struct student a = {"Alice", 82, 72, 58};
    struct student b = {"Bob"};
    b.english = 71;
    b.math = 92;
    b.physics = 98;

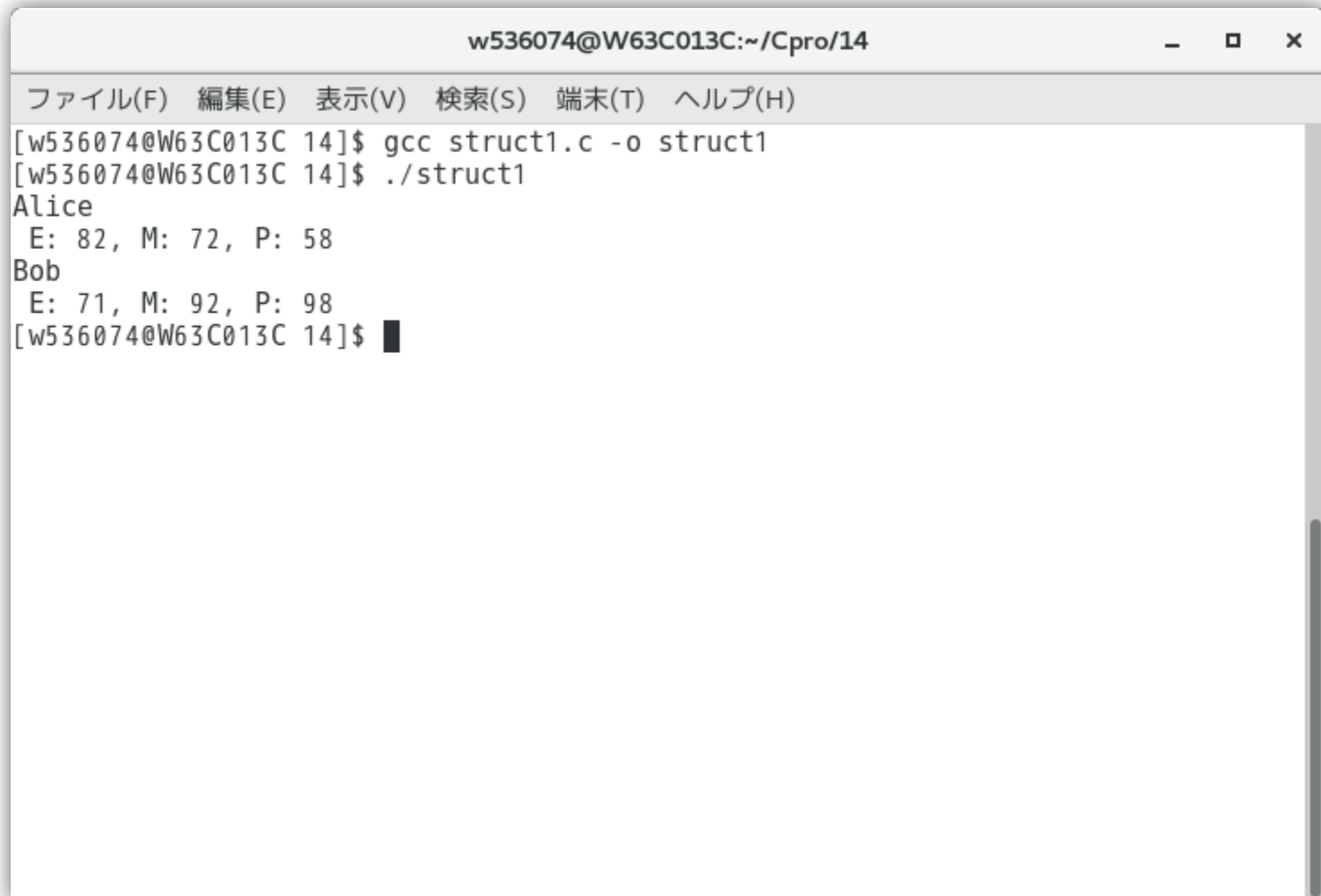
    printf("%s\n E: %d, M: %d, P: %d\n",  );
    printf("%s\n E: %d, M: %d, P: %d\n",  );
    return 0;
}
```

... 前スライドまでと意味は一緒

名前と各科目の点数を表示
(2人分)



演習① ~実行結果~



```
w536074@W63C013C:~/Cpro/14
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C013C 14]$ gcc struct1.c -o struct1
[w536074@W63C013C 14]$ ./struct1
Alice
E: 82, M: 72, P: 58
Bob
E: 71, M: 92, P: 98
[w536074@W63C013C 14]$
```

演習②

- ▶ 演習①のプログラムを改良し，各学生の平均点を算出するプログラム struct2.c を作成せよ。

```
#include <stdio.h>
```

```
struct student{  
    ...(省略)...  
};
```

?

(calc_avg関数を自作する)

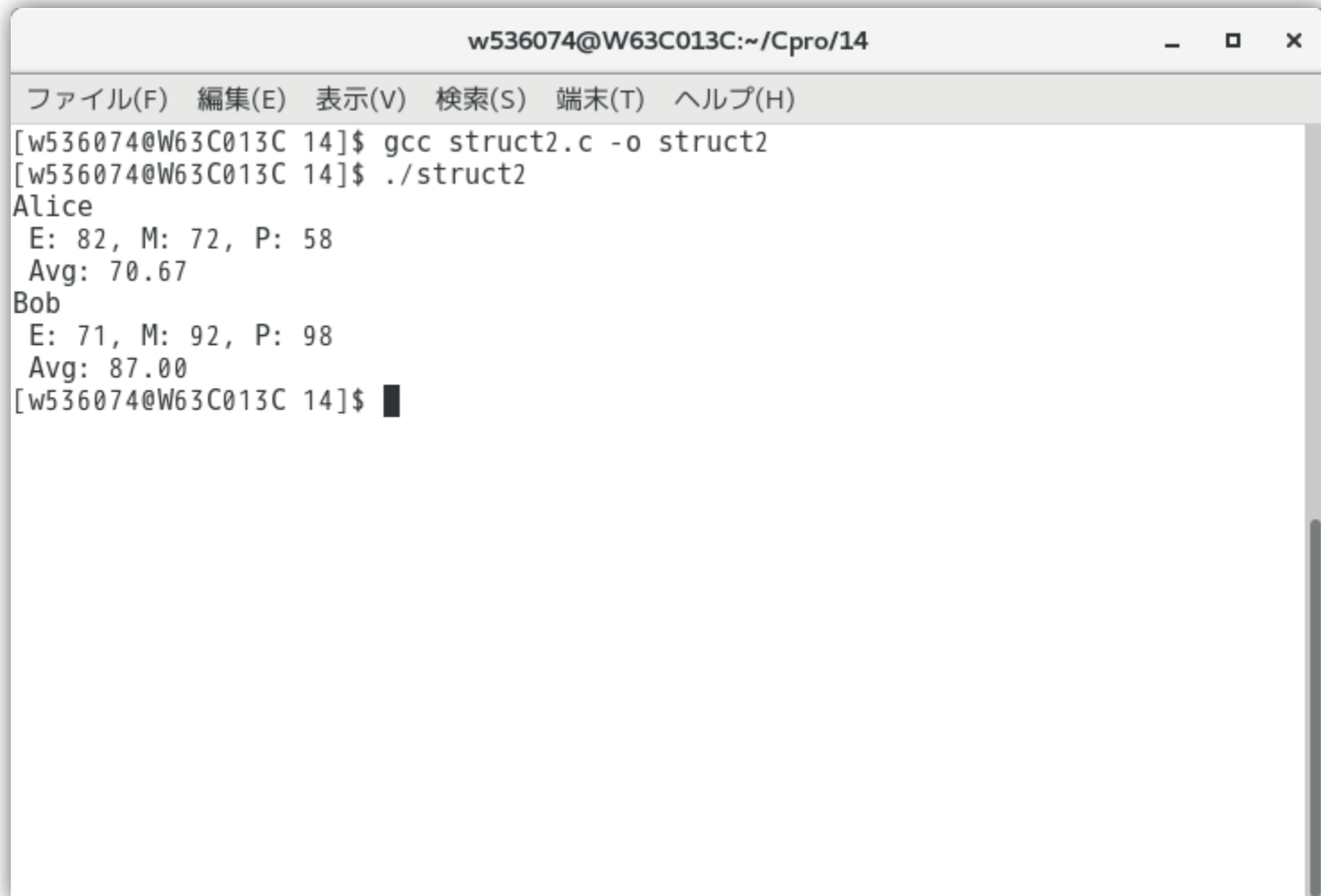
```
int main(void){  
    ...(省略)...  
  
    a.avg = calc_avg(a.english, a.math, a.physics);  
    b.avg = calc_avg(b.english, b.math, b.physics);  
  
    printf("%s\n E: %d, M: %d, P: %d\n",  
    printf(" Avg: %.2f\n",  
    printf("%s\n E: %d, M: %d, P: %d\n",  
    printf(" Avg: %.2f\n",  
    return 0;  
}
```

三科目の点数から平均点を算出
(2人分)



```
    printf("%s\n E: %d, M: %d, P: %d\n",  
    printf(" Avg: %.2f\n",  
    printf("%s\n E: %d, M: %d, P: %d\n",  
    printf(" Avg: %.2f\n",  
    return 0;
```


演習② ~実行結果~



```
w536074@W63C013C:~/Cpro/14
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C013C 14]$ gcc struct2.c -o struct2
[w536074@W63C013C 14]$ ./struct2
Alice
E: 82, M: 72, P: 58
Avg: 70.67
Bob
E: 71, M: 92, P: 98
Avg: 87.00
[w536074@W63C013C 14]$
```

演習③

- ▶ 演習②のプログラムを改良し，各学生に成績を付与するプログラム struct3.c を作成せよ。

```
#include <stdio.h>
```

```
struct student{  
    ...(省略)...  
};
```

?

(check関数を自作する)

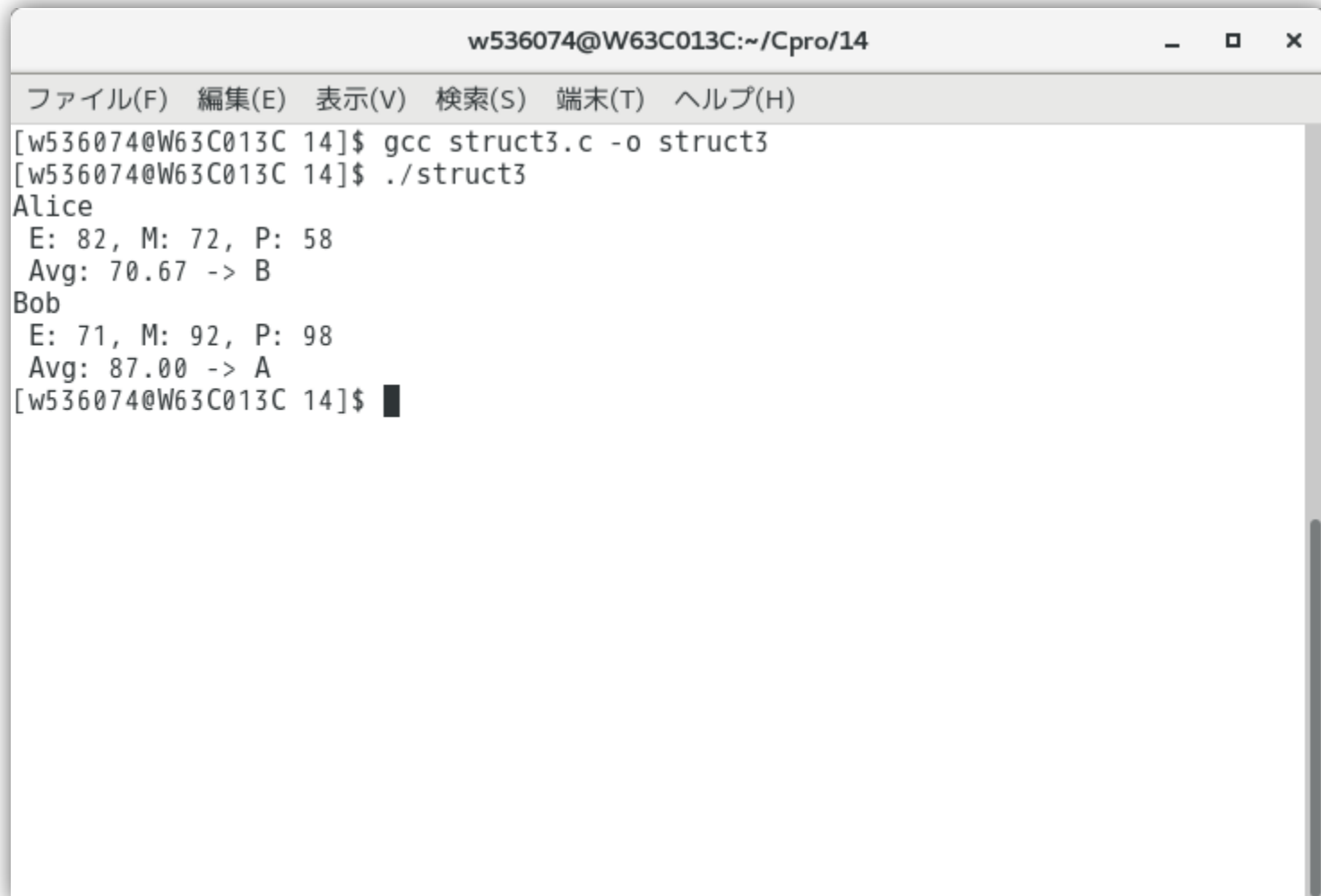
calc_avg関数は
残しておく

```
int main(void){  
    ...(省略)...  
  
    a.grade = check(a.avg);  
    b.grade = check(b.avg);  
  
    printf("%s\n E: %d, M: %d, P: %d\n",  
    printf(" Avg: %.2f -> %c\n",  
    printf("%s\n E: %d, M: %d, P: %d\n",  
    printf(" Avg: %.2f -> %c\n",  
    return 0;  
}
```

平均点に応じて成績評価
(2人分)

平均点	成績
90点以上	S
80点以上90点未満	A
70点以上80点未満	B
60点以上70点未満	C
60点未満	D

演習③ ~実行結果~



```
w536074@W63C013C:~/Cpro/14
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C013C 14]$ gcc struct3.c -o struct3
[w536074@W63C013C 14]$ ./struct3
Alice
E: 82, M: 72, P: 58
Avg: 70.67 -> B
Bob
E: 71, M: 92, P: 98
Avg: 87.00 -> A
[w536074@W63C013C 14]$
```

演習④

- ▶ 演習③のプログラムを改良し、新たに一人の学生を追加するプログラム struct4.c を作成せよ。
 - ▶ ただし、追加の学生については名前と各科目の点数を端末から入力できるようにすること
- ▶ 実行例

```
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C013C 14]$ gcc struct4.c -o struct4
[w536074@W63C013C 14]$ ./struct4
Your name: Carol
E score: 38
M score: 56
P score: 12
Alice
  E: 82, M: 72, P: 58
  Avg: 70.67 -> B
Bob
  E: 71, M: 92, P: 98
  Avg: 87.00 -> A
Carol
  E: 38, M: 56, P: 12
  Avg: 35.33 -> F
```

} 端末から入力

演習⑤

- ▶ 複素数を表現する構造体を定義し、複素数の加算を行なう関数を作成せよ。
- ▶ 次スライドにプログラム例を示す。
 - ▶ 講義中であれば, **cp /share/complex_add.c ./** で取得できる

演習⑤

```
#include <stdio.h>

struct my_complex{
    double re;      ... 実部 (Real part)
    double im;      ... 虚部 (Imaginary part)
};

struct my_complex complex_add(struct my_complex x, struct my_complex y){
    

?

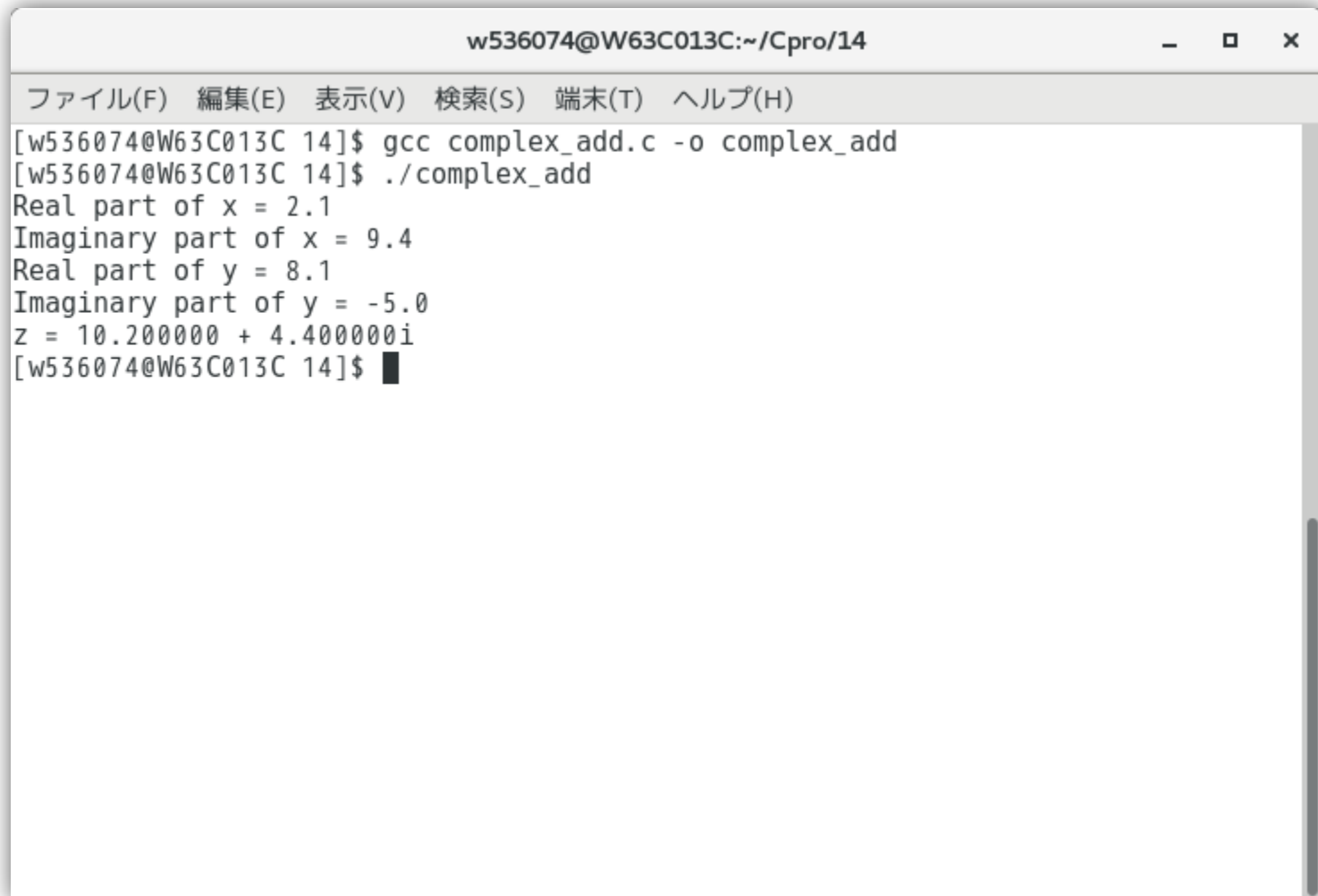

}

int main(void){
    struct my_complex x, y, z;
    printf("Real part of x = "); scanf("%lf", &x.re);
    printf("Imaginary part of x = "); scanf("%lf", &y.im);
    printf("Real part of y = "); scanf("%lf", &y.re);
    printf("Imaginary part of y = "); scanf("%lf", &y.im);

    z = complex_add(x, y);    ... 複素数の加算 (引数と戻り値が構造体)

    printf("z = %f + %fi\n", z.re, z.im);
    return 0;
}
```

演習⑤ ~実行結果~



```
w536074@W63C013C:~/Cpro/14
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[w536074@W63C013C 14]$ gcc complex_add.c -o complex_add
[w536074@W63C013C 14]$ ./complex_add
Real part of x = 2.1
Imaginary part of x = 9.4
Real part of y = 8.1
Imaginary part of y = -5.0
z = 10.200000 + 4.400000i
[w536074@W63C013C 14]$
```

キーワード

- ▶ 本日のキーワード：
- ▶ 本日**最終回**
- ▶ 退出前に**アンケート**を提出してください。
 - ▶ 設問Ⅰ～設問Ⅲに回答