(12-1) Recursion
H&K Chapter 9

Instructor – Beiyu Lin
CptS 121 (June 4th, 2019)
Washington State University

WASHINGTON STATE UNIVERSITY
World Class. Face to Face.

Review Struct

```
/* in the struct.h file */

typedef enum {freshman, sophomore, junior, senior} class_t;
typedef enum {anthropology, biology, chemistry,
              english, compsci, polisci,psychology, physics,
              engineering, sociology} major_t;

typedef struct
{
     int id_number;
     class_t class_standing; /* see above */
     major_t major; /* see above */
     double gpa;
    int credits_taken;
} student_t;
```


C. Hundhausen, A. O'Fallon, B. Lin

struct Type (8)

- Here's how we could use the previous function:

```
/* in the main.c file */
int main(void)
{
   student_t student1, student2;
   read_student(&student1);
   read_student(&student2);
   print_student(student1); /* assume print_student is defined */
   print_student(student2);
   return(1);
}
```

Similar as read in data from file

C. Hundhausen, A. O'Fallon, B. Lin

Review Struct

```
/*in the functions.c file*/

void read_student(student_t *student)
{   int temp_class, temp_major;
    printf("Please enter ID number of student: ");
    scanf("%d",&(*student).id_num);

    printf("Please enter class standing (0 = fr,\n");
    printf("1 = so, 2 = ju, 3 = se): ");
    scanf("%d",&temp_class);
    (*student).class = (class_t)temp_class;

    printf("Please enter major (0 = anthro.,\n");
    printf("1 = biol., 2 = chem., … , 8 = soc.: ");
    scanf("%d",&temp_major);
    (*student).major = (major_t)temp_major;

    printf("Please enter gpa: ");
    scanf("%lf",&(*student).gpa);

    }
```


C. Hundhausen, A. O'Fallon, B. Lin

Review Struct

How to check if the read in information?

- printf("the id number is %d \n", student1.id_number);

How about the class standing? Freshman, sophomore…?

Use **if or Switch**

- If (student1.class_standing == freshman) {printf("a freshman\n");
- switch(student1.class_standing)

```
{
 case freshman:
        printf("A freshman \n");
        break;
 …..
}
```


C. Hundhausen, A. O'Fallon, B. Lin

What is a Recursive Function?

Graphs are from:
https://www.banggood.com/Set-of-5-Cute-Wooden-Nesting-Dolls-Matryoshka-Animal-Russian-Doll-p-964569.html?cur_warehouse=CN
https://www.imdb.com/title/tt7520794/

C. Hundhausen, A. O'Fallon, B. Lin

What is a Recursive Function?

- A function that calls itself either directly or indirectly through another function
  - For example:

```
int recursive_function (int r, int s)
{
    …
    recursive_function (r, s-1) /* recursive call */
    …
}
```


C. Hundhausen, A. O'Fallon

Nature of Recursion (1)

- Problems that may be solved using recursion have these attributes:
  - One or more simple cases have a straightforward, non-recursive solution
  - The other cases may be defined in terms of problems that are closer to the simple cases
  - Through a series of calls to the recursive function, the problem eventually is stated in terms of the simple cases

C. Hundhausen, A. O'Fallon

Nature of Recursion (2)

- As described by Wirth:

  "The power of recursion evidently lies in the possibility of defining an infinite set of objects by a finite statement. In the same manner, an infinite number of computations can be described by a finite recursive program, even if this program contains no explicit repetitions."

C. Hundhausen, A. O'Fallon

Nature of Recursion (3)

- A divide and conquer approach
- A fresh copy of a function goes to work on a similar, but simpler problem than the original

C. Hundhausen, A. O'Fallon

Properties of Recursion

- Recursive solutions have at least two cases:
  - The simple or *base* case(s)
  - The *recursive* case(s) or step(s)
- Note:

Any problem can be solved by recursion can also be resolved by iteration.

C. Hundhausen, A. O'Fallon, B. Lin

Properties of Recursion

- Key differences

	Recursion	v.s.	Iteration
Infinite	system crash		consumes CPU cycles.
processor time	expensive		Not
memory space	expensive		Not
code	smaller		longer

C. Hundhausen, A. O'Fallon, B. Lin

Example of Recursive Solution

- This is an example provided on p. 528 of the Hanly & Koffman text. This example performs multiplication through addition.

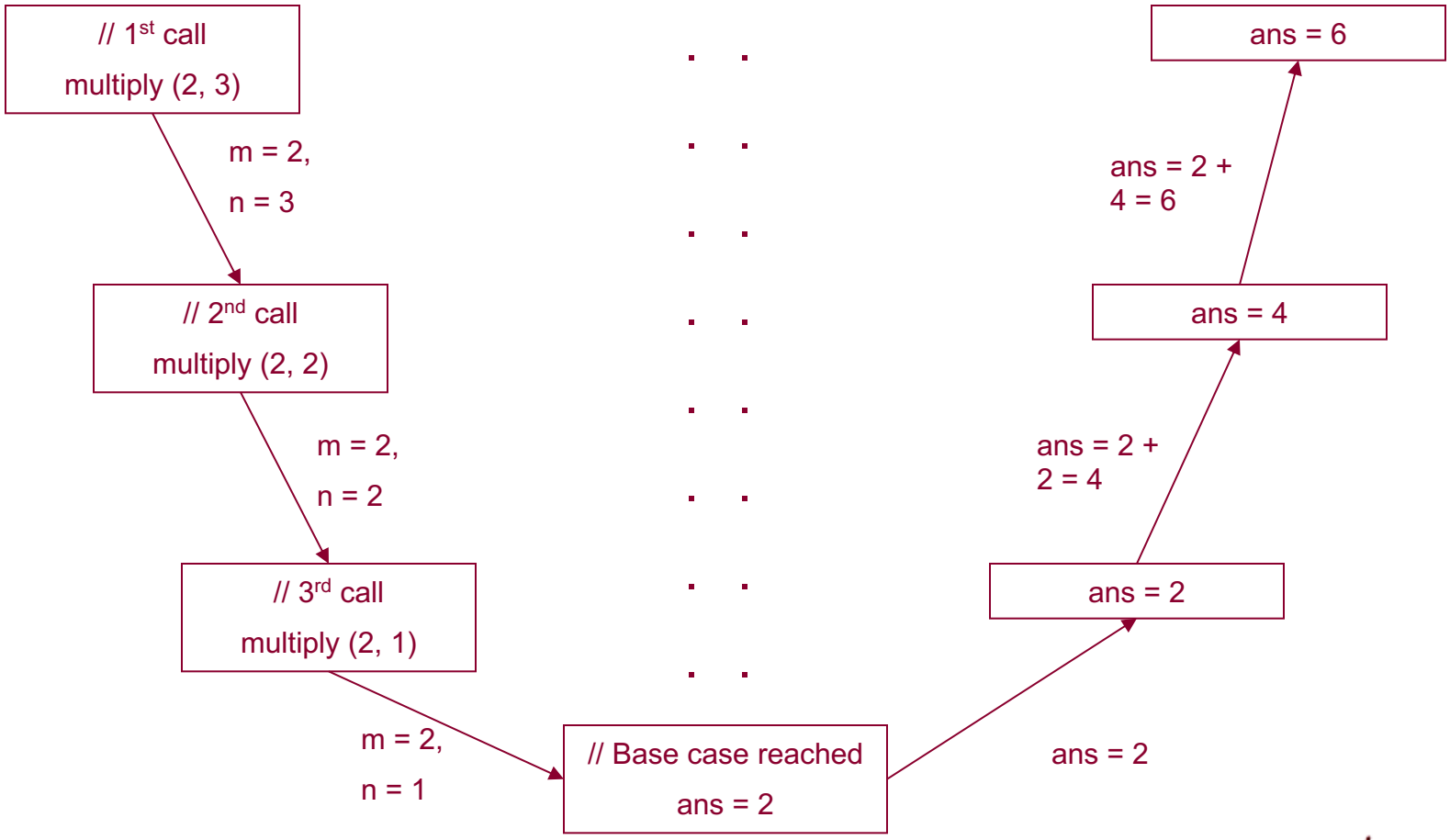
```
int multiply (int m, int n)
{
        int ans;

        if (n == 1)
        {
                ans = m; /* simple or base case */
        }
        else
        {
                ans = m + multiply (m, n – 1); /* recursive step */
        }

        return ans;
}
```


C. Hundhausen, A. O'Fallon

Evaluation of Recursive Multiply



C. Hundhausen, A. O'Fallon

Next Lecture…

- More recursive examples

References

- J.R. Hanly & E.B. Koffman, *Problem Solving and Program Design in C (8th Ed.)*, Addison-Wesley, 2016
- P.J. Deitel & H.M. Deitel, *C How to Program (7h Ed.)*, Pearson Education , Inc., 2013.
- N. Wirth, *Algorithms + Data Structures = Programs*, Prentice-Hall, 1976

Collaborators

- Chris Hundhausen
- Andrew O'Fallon