CME 112- Programming Languages II

Week 3
Examples with
Recursive Functions

Assist. Prof. Dr. Caner Özcan

Without conscience, knowledge only hurts the soul. ~ Rabelais

Recursion Factorial

► Recursion program to calculate and print the factorials of integers 0-10.

```c
#include <stdio.h>
long faktorial(long n){
    if (n <= 1)
        return 1;
    else
        return (n*faktorial(n - 1));
}
int main(void){
    int i;
    for (i = 0; i <= 10; i++) {
        printf("%d! = %d\n", i, faktorial(i));
    }
    return 0;
}
```

Recursion Fibonacci Numbers

► Calculating nth Fibonacci number recursively.

```c
#include <stdio.h>
long fibonacci(long n){
    if (n == 0 || n == 1)
        return n;
    else
        return fibonacci(n - 1) + fibonacci(n - 2);
}
int main(void){
    long i, n;
    printf("How many fibonacci numbers?:");
    scanf("%d", &n);
    for (i = 1; i <= n; i++){
        printf("Number %d: %ld\n", i, fibonacci(i));
    }
    return 0;
}
```

Recursion

► Find out the output of the program below. What does fun() do in general?

```c
#include <stdio.h>

int fun(int a[], int n){
        int x;
        if (n == 1)
                return a[0]; //base case – temel durum
        else
                x = fun(a, n - 1);
        if (x > a[n - 1])
                return x;
        else
                return a[n - 1];
}

int main(){
        int arr[] = { 12, 10, 300, 50, 100 };
        printf(" %d ", fun(arr, 5));
        getchar();
        return 0;
}
```

Recursion

a={ 12, 10, 300, 50, 100 }

Starting Point

fun(int a[], int 5)	n=5
x = fun(a, n - 1)	n=4

fun(int a[], int 4)	n=4
x = fun(a, n - 1)	n=3

fun(int a[], int 3)	n=3
x = fun(a, n - 1)	n=2

fun(int a[], int 2)	n=2
x = fun(a, n - 1)	n=1

fun(int a[], int 1)	n=1
return a[0]	

12 base case

```c
int fun(int a[], int n)
{
int x;

if (n == 1)
    return a[0]; //base case - temel durum
else
    x = fun(a, n - 1);
if (x > a[n - 1])
    return x;
else
    return a[n - 1];
}
```

Recursion

a={ 12, 10, 300, 50, 100 }

```
int fun(int a[], int n)
{
int x;

if (n == 1)
    return a[0]; //base case - temel durum
else
    x = fun(a, n - 1);
if (x > a[n - 1])
    return x;
else
    return a[n - 1];
}
```

Starting Point

```
if (x > a[5 - 1])
```

| fun(int a[], int 5) | n=5 |
| x = fun(a, n - 1) | n=4 |

300

End Point

```
if (x > a[4 - 1])
```

300

| fun(int a[], int 4) | n=4 |
| x = fun(a, n - 1) | n=3 |

```
if (x > a[3 - 1])
```

300

| fun(int a[], int 3) | n=3 |
| x = fun(a, n - 1) | n=2 |

```
if (x > a[2 - 1])
```

12

| fun(int a[], int 2) | n=2 |
| x = fun(a, n - 1) | n=1 |

| fun(int a[], int 1) | n=1 |
| return a[0] | |
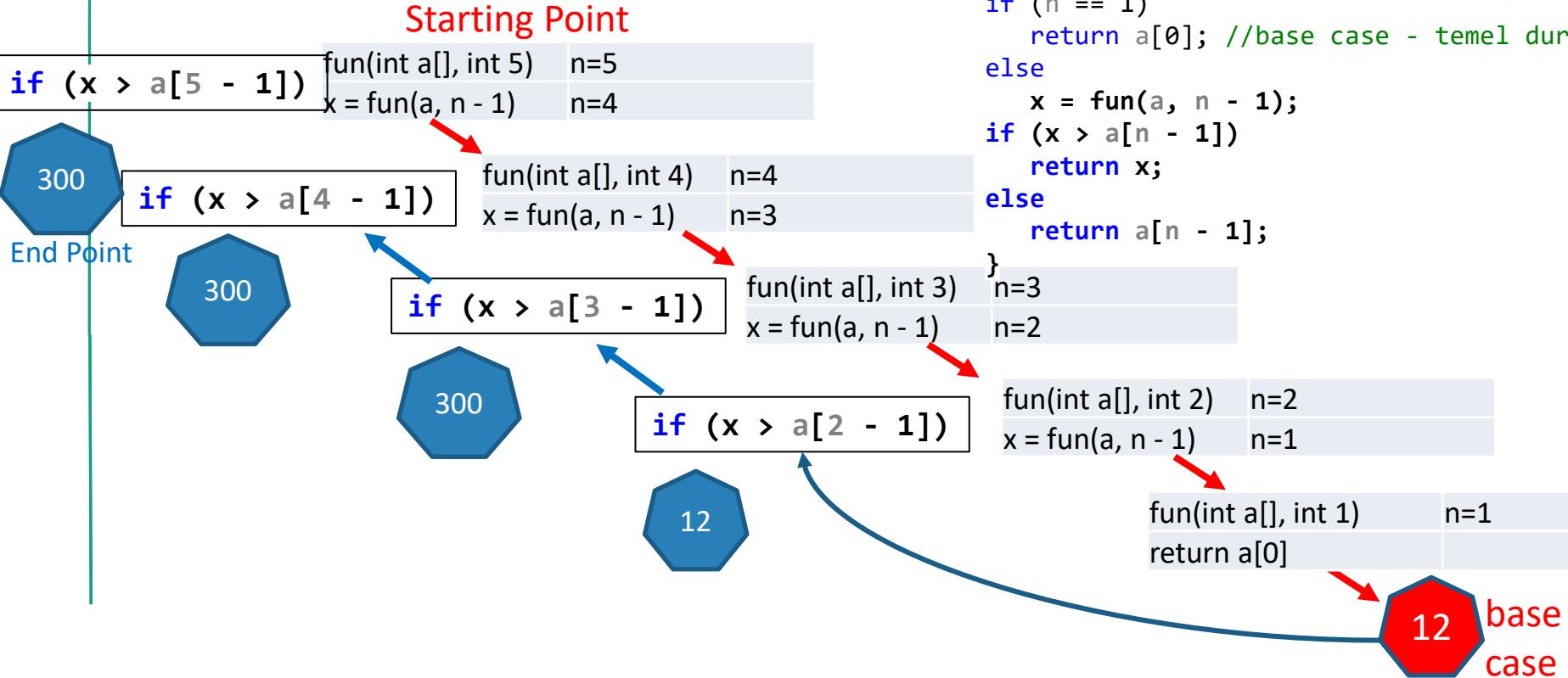
12 base case

Recursion

```c
#include <stdio.h>
int fun1(int x, int y){
        if (x == 0)
                return y;//base case – temel durum
        else
                return fun1(x - 1, x + y);
}
void fun2(int n){
        int i = 0;
        if (n > 1)            {
                fun2(n - 1);
                printf("\n");
        }
        for (i = 0; i < n; i++) //base case – temel durum
                printf(" * ");
}
int main(){
        int a = 5, b = 2,c;
        c = fun1(a, b);
        printf("Value from the function: %d\n",c);
        fun2(5);
        getchar();
}
```

Recursion

```c
#include <stdio.h>

int SumOfElement(int[], int);

int main(){
        int n=5 ,dizi[5]={3,7,2,9,1};
        int toplam=SumOfElement(dizi,n-1);
        printf("Toplam = %d",toplam);
        return 0;
 }

int SumOfElement(int dizi[], int n){
        if(n==0)
                return dizi[0];
        return dizi[n]+SumOfElement(dizi,n-1);
}
```

Recursion

```c
#include <stdio.h>
unsigned int multiply(unsigned int x, unsigned int y){
    if (x == 1){
        /* Terminating case */
        return y;
    }
    else if (x > 1){
        /* Recursive step */
        return y + multiply(x-1, y);
    }
    /* Catch scenario when x is zero */
    return 0;
}
int main() {
    printf("7 times 5 is %d", multiply(7, 5));
    return 0;
}
```

Recursion

```c
#include <stdio.h>
int sum (int num){
    if (num != 0){
        return (num % 10 + sum (num / 10));
    }
    else{
        return 0;
    }
}
int main(){
    int num, result;

    printf("Enter the number: ");
    scanf("%d", &num);
    result = sum(num);
    printf("Sum of digits in %d is %d\n", num, result);
    return 0;
}
```

QUIZ

Write screen output, when **fun** is called in main function with **n=10, a=5 and b=7**.
(**fun** fonksiyonu main fonksiyondan **n=10, a=5 ve b=7** değeri için çağırıldığındaki ekran çıktısını yazın.)

```c
void fun(int n, int a, int b){

        if (n <= 0)

                return;

        fun(n - 2, a, b + n);

        printf("%d %d %d\n", n, a, b);

}
```

QUIZ

Write a recursive function that finds the given power of a given number.
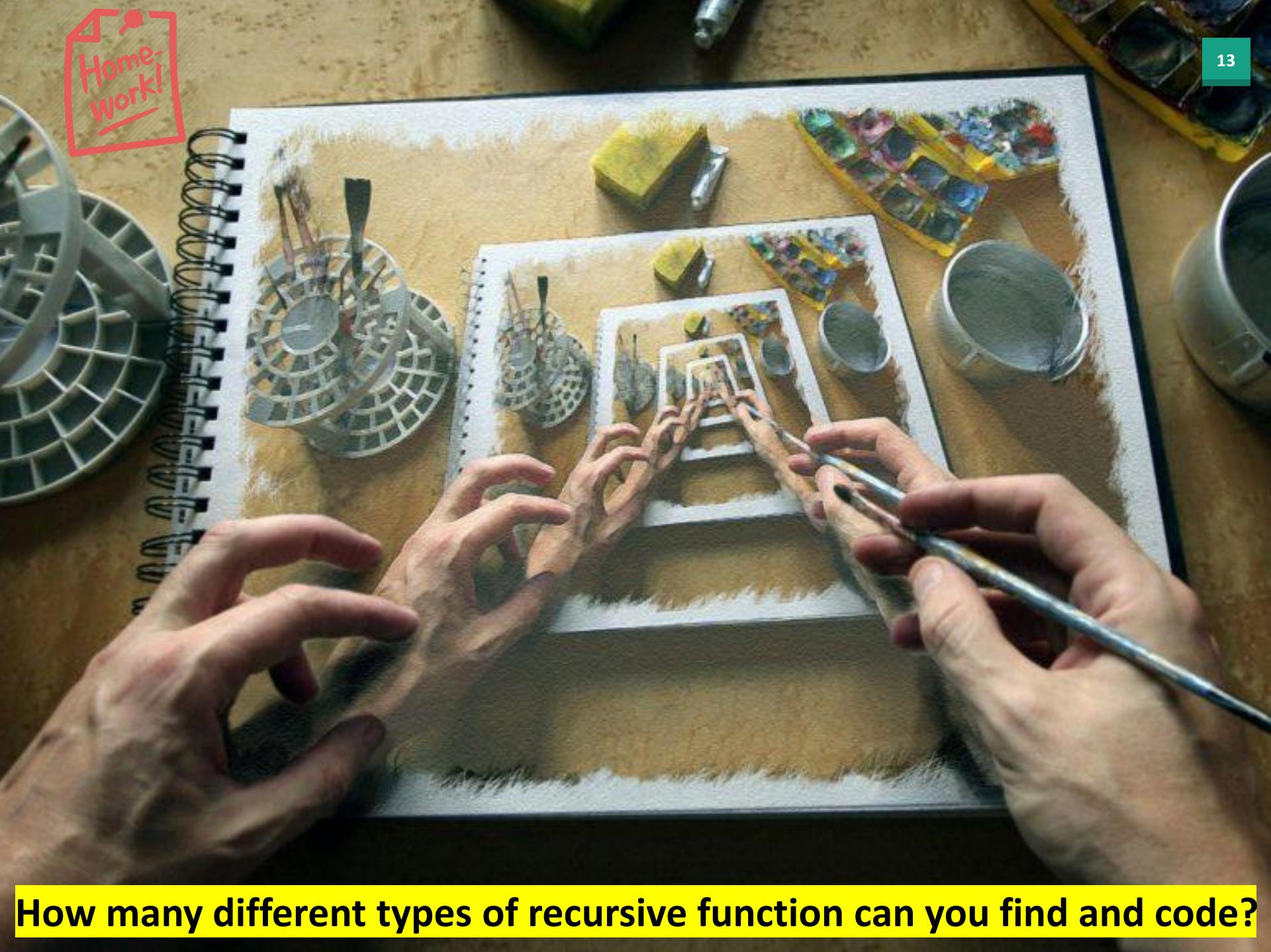
How many different types of recursive function can you find and code?

Next Week

▶ Pointers

▶ Call by Value

▶ Call by Reference

▶ Dynamic Memory Allocation

References

▶ Doç. Dr. Fahri Vatansever, "Algoritma Geliştirme ve Programlamaya Giriş", Seçkin Yayıncılık, 12. Baskı, 2015.

▶ Kaan Aslan, "A'dan Z'ye C Klavuzu 8. Basım", Pusula Yayıncılık, 2002.

▶ Paul J. Deitel, "C How to Program", Harvey Deitel.

▶ "A book on C", All Kelley, İra Pohl

Any
Questions
?

Thanks for listening

CANER ÖZCAN
canerozcan@karabuk.edu.tr

Hayatta en pahalı şey tecrübedir,
çünkü kazanmak için kaybetmek gerekir.