

Programming with C I

Fangtian Zhong
CSCI 112

Gianforte School of Computing
Norm Asbjornson College of Engineering
E-mail: fangtian.zhong@montana.edu

Array Arguments

- **We can write functions that have arrays as arguments.**
- **Such functions can manipulate some, or all, of the elements corresponding to an actual array argument.**

Using Array Elements as Function Arguments

```
scanf("%lf", &x[i]);
```

Figure Function to Check Whether Tic-tac-toe Board is Filled

```
/* Check Whether a tic-tac-toe is completely filled.          */
int filled(char ttt_brd[3][3])    /* input -tic-tac-toe board */
{
    int r, c;    /* row and column subscripts    */
    int ans=1;    /* whether or not board filled    */

    / * Assumes board is filled until blank is found          */
    for (r = 0; r < 3; ++r)
        for (c = 0; c < 3; ++c)
            if (ttt_brd[r][c])
                ans = 0;

    return (ans);
}
```

Variable scope

- **Part of a program where a variable is accessible**
- **Lifetime of a variable**

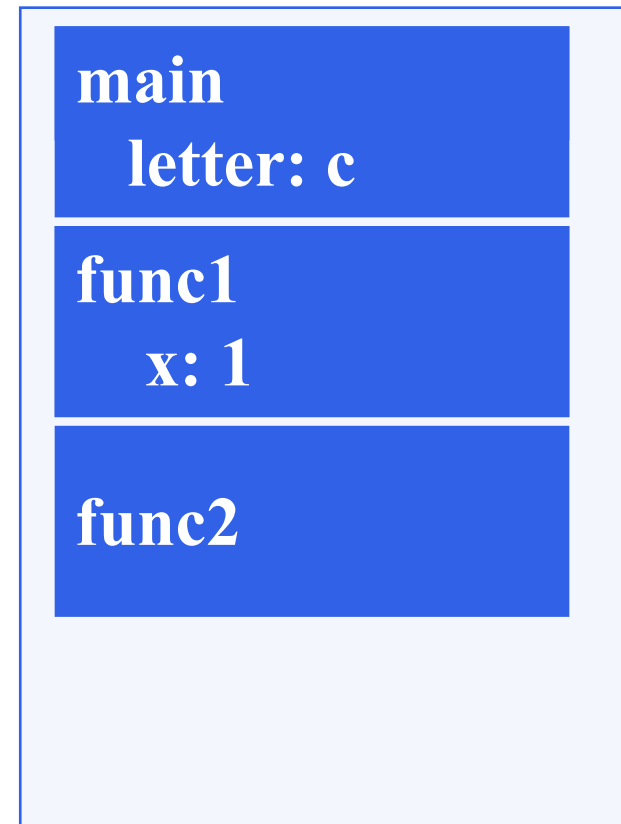
What happens when we run our executable file?

```
func2() {  
    printf("%d\n", x);  
}  
func1() {  
    int x = 1;  
    func2();  
}  
int main(void) {  
    char letter='c'  
    func1();  
}
```

out of scope!



Memory



What happens when we run our executable file?

```
void fill_array(  
    int list[],  
    int n,  
    int in_value) {  
    int i;  
    for (i = 0;  
        i < n; ++i) {  
        list[i] = in_value;  
    }  
}  
  
int main(void) {  
    int arr[10];  
    fill_array(arr, 5, 1);  
}
```

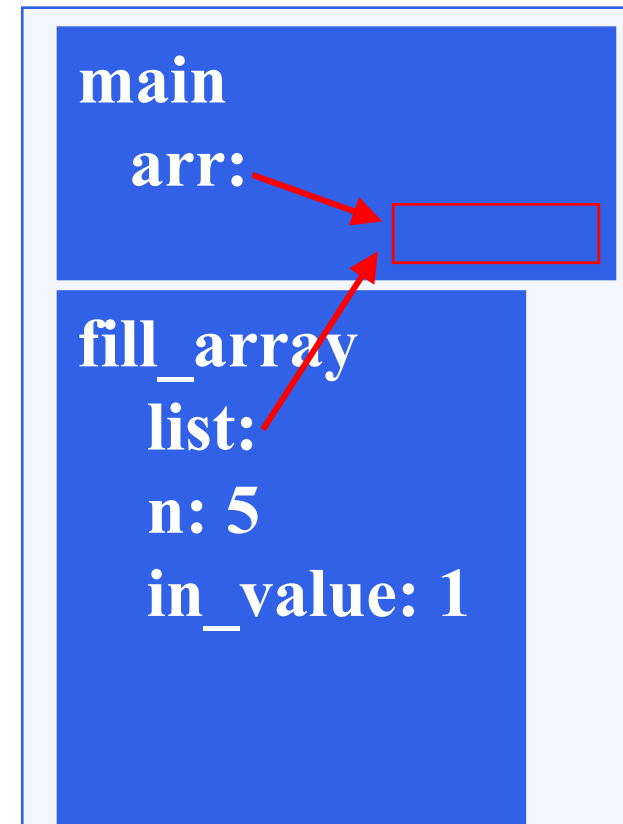
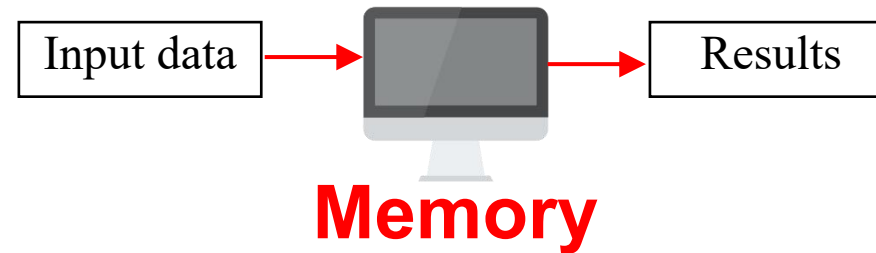
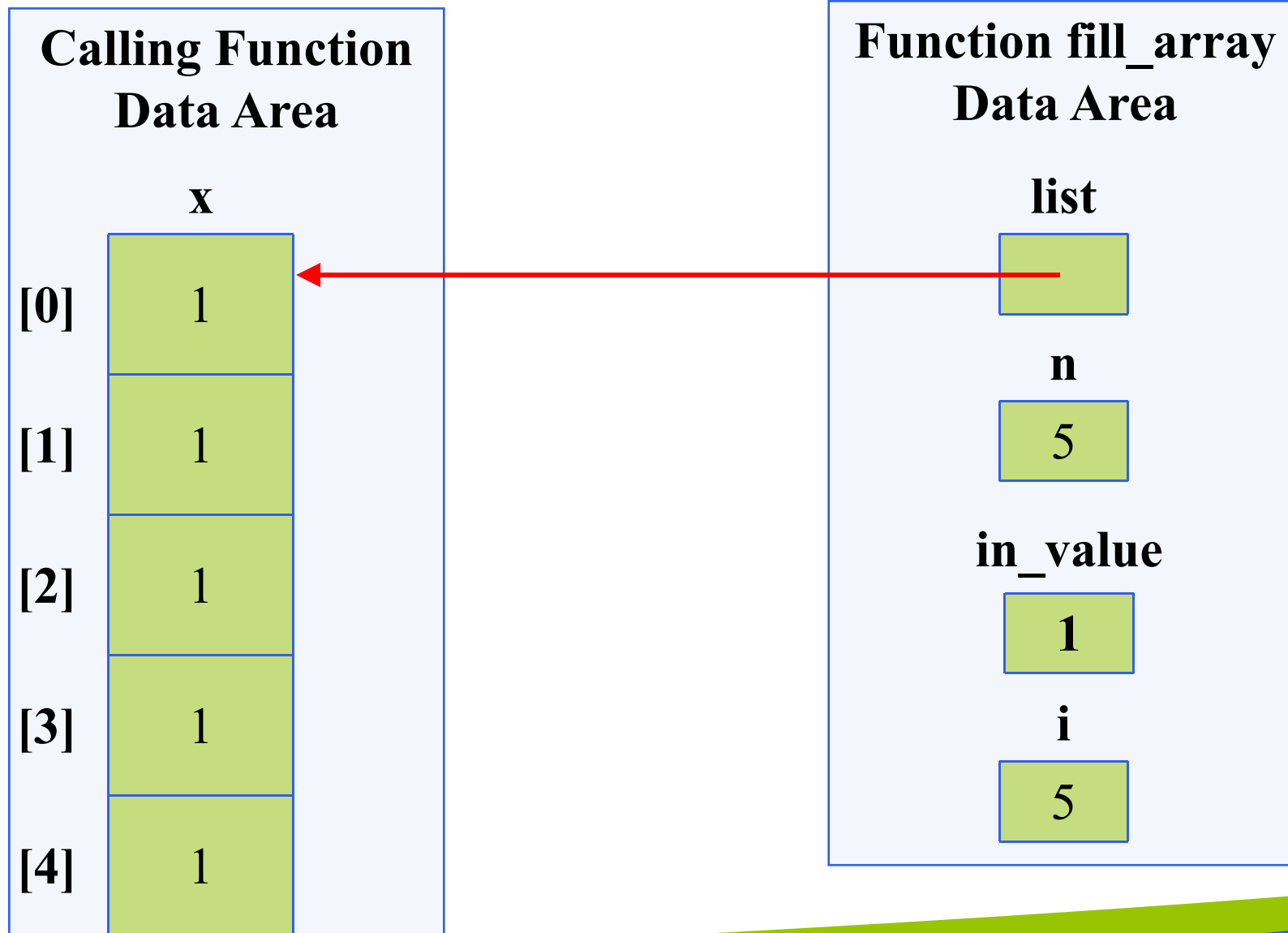


Figure Function fill_array

```
/*
 * Set all elements of its array parameter to in_value.
 * Pre: n and in_value are defined.
 * Post: list[i] = in_value, for  $0 \leq i < n$ .
 */
void
fill_array (int list[],      /* output - list of n integers      */
            int n,          /* input - number of list elements */
            int in_value)   /* input - initial value          */
{
    for (int i = 0; i < n; ++i)
        list[i] = in_value;
}
```


Figure Data Areas Before Return from fill_array (x, 5, 1);



Arrays as Input Arguments


 The qualifier **const** allows the compiler to mark as an error any attempt to change an array element within the function.

Figure Function to Find the Largest Element in an Array

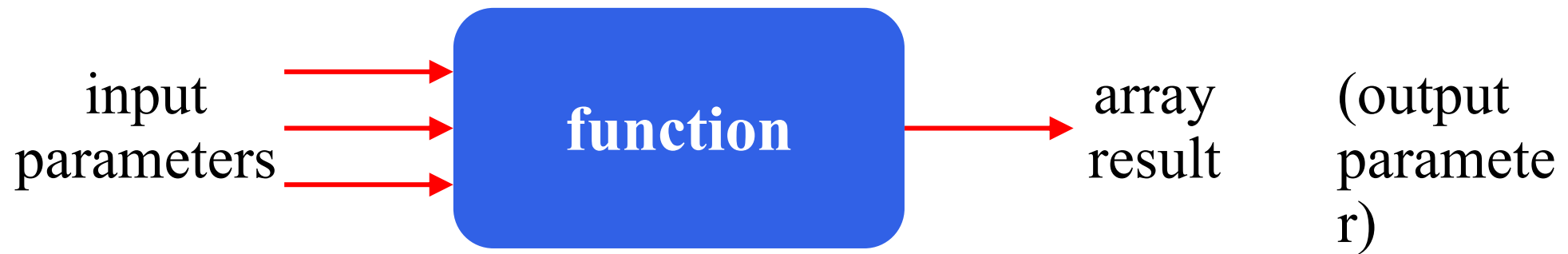
```
/*
 * Return the largest of the first n values in array list
 * Pre: First n elements of array list are defined and  $n > 0$ 
 */
int
get_max(const int list[], /* input - list of n integers */
        int n) /* input - number of list elements to examine */
{
    int cur_large; /* largest value so far */

    /* Initial array element is largest so far */
    cur_large = list[0];

    /* Compare each remaining list element to the largest so far;
       save the larger
    for (int i = 1; i < n; ++i){
        if (list[i] > cur_large)
            cur_large = list[i]
    }
    return (cur_large);
}
```

Returning an Array Result

- 🛡️ In C, it is not legal for a function's return type to be an array.
- 🛡️ You need to use an output parameter to send your array back to the calling module.

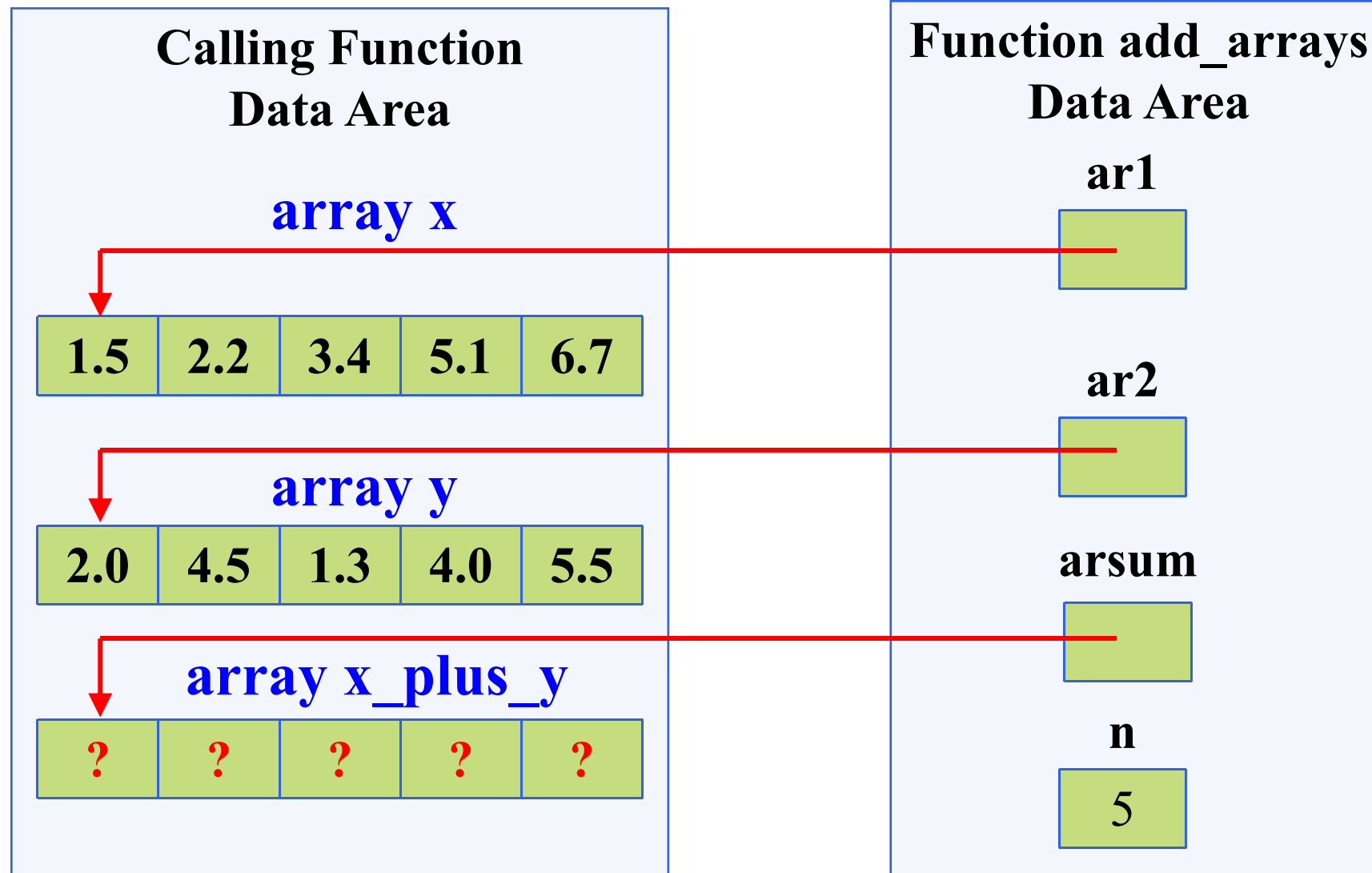


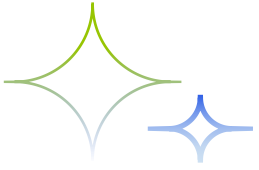
➤ **Diagram of a function That Computes an Array Result**

Figure Function to Add Two Arrays

```
/*
 * Adds corresponding elements of arrays ar1 and ar2, storing the result in arsum.
 * Processes first n elements only.
 * Pre: First n elements of ar1 and ar2 are defined. arsum's corresponding actual
       argument has a declared size  $\geq n$  ( $n \geq 0$ )
 */
void add_arrays(const double ar1[],          /* input - */
               const double ar2[],          /* arrays being added */
               double arsum[],              /* output - sum of ar1 and ar2 */
               int n)                      /* input - number of element paris summed*/
{
    /* Adds corresponing elements of ar1 and ar2 */
    for (int i = 0; i < n; ++i)
        arsum[i] = ar[i] + ar2[i];
}
```

Figure Function Data Areas for add_arrays(x, y, x_plus_y, 5);





THE END

Fangtian Zhong
CSCI 112

Gianforte School of Computing
Norm Asbjornson College of Engineering
E-mail: fangtian.zhong@montana.edu