C Structure and Function

In this tutorial, you'll learn to pass struct variables as arguments to a function. You will learn to return struct from a function with the help of examples.

Similar to variables of built-in types, you can also pass structure variables to a function.

Passing structs to functions

We recommended you to learn these tutorials before you learn how to pass structs to functions.

- C structures
- C functions
- User-defined Function

Here's how you can pass structures to a function

```
#include <stdio.h>
struct student {
   char name[50];
   int age;
};

// function prototype
void display(struct student s);

int main() {
   struct student s1;

   printf("Enter name: ");

   // read string input from the user until \n is entered
```

```
    // \n is discarded
    scanf("%[^\n]%*c", s1.name);

    printf("Enter age: ");
    scanf("%d", &s1.age);

    display(s1); // passing struct as an argument

    return 0;
}

void display(struct student s) {
    printf("\nDisplaying information\n");
    printf("Name: %s", s.name);
    printf("\nAge: %d", s.age);
}
```

Output

```
Enter name: Bond
Enter age: 13

Displaying information
Name: Bond
Age: 13
```

Here, a struct variable `s1` of type `struct student` is created. The variable is passed to the `display()` function using `display(s1);` statement.

Return struct from a function

Here's how you can return structure from a function:

```
#include <stdio.h>
struct student
{
    char name[50];
    int age;
};
```

```c
// function prototype
struct student getInformation();

int main()
{
    struct student s;

    s = getInformation();

    printf("\nDisplaying information\n");
    printf("Name: %s", s.name);
    printf("\nRoll: %d", s.age);

    return 0;
}
struct student getInformation()
{
  struct student s1;

  printf("Enter name: ");
  scanf ("%[^\n]%*c", s1.name);

  printf("Enter age: ");
  scanf("%d", &s1.age);

  return s1;
}
```

Here, the `getInformation()` function is called using `s = getInformation();` statement. The function returns a structure of type `struct student`. The returned structure is displayed from the `main()` function.

Notice that, the return type of `getInformation()` is also `struct student`.

Passing struct by reference

You can also pass structs by reference (in a similar way like you pass variables of built-in type by reference). We suggest you to read pass by reference tutorial before you proceed.

During pass by reference, the memory addresses of struct variables are passed to the function.

```c
#include <stdio.h>
typedef struct Complex
{
    float real;
    float imag;
} complex;

void addNumbers(complex c1, complex c2, complex *result);

int main()
{
    complex c1, c2, result;

    printf("For first number,\n");
    printf("Enter real part: ");
    scanf("%f", &c1.real);
    printf("Enter imaginary part: ");
    scanf("%f", &c1.imag);

    printf("For second number, \n");
    printf("Enter real part: ");
    scanf("%f", &c2.real);
    printf("Enter imaginary part: ");
    scanf("%f", &c2.imag);

    addNumbers(c1, c2, &result);
    printf("\nresult.real = %.1f\n", result.real);
    printf("result.imag = %.1f", result.imag);

    return 0;
}
void addNumbers(complex c1, complex c2, complex *result)
{
     result->real = c1.real + c2.real;
     result->imag = c1.imag + c2.imag;
}
```

Output

```
For first number,
Enter real part:  1.1
```

```
Enter imaginary part:  -2.4
For second number,
Enter real part:  3.4
Enter imaginary part:  -3.2

result.real = 4.5
result.imag = -5.6
```

In the above program, three structure variables `c1`, `c2` and the address of `result` is passed to the `addNumbers()` function. Here, `result` is passed by reference.

When the `result` variable inside the `addNumbers()` is altered, the `result` variable inside the `main()` function is also altered accordingly.