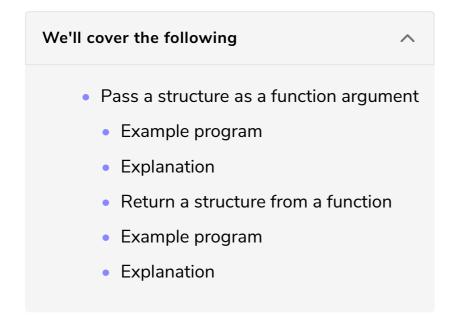
### Structure and Functions

Let's see how to pass a structure as an argument to the function.



# Pass a structure as a function argument #

In the previous lesson, we saw that printing the members of each structure variable is a repetitive task. So can we define a function in which we just pass the structure variable, and it prints the values for us? Yes, we can.

## Example program #

Press the **RUN** button and see the output!

```
#include <iostream>

using namespace std;

// Student structure
struct Student {
   string name;
   int roll_number;
```

```
int marks;
};
// printStudent function
void printStudent(Student s) {
  cout << "Student information" << endl;</pre>
  cout << "Name = " << s.name << endl;</pre>
  cout << "Roll Number = " << s.roll_number << endl;</pre>
  cout << "Marks = " << s.marks << endl;</pre>
// main function
int main() {
  struct Student s[100];
  s[0] = { "John", 1, 50 };
  printStudent(s[0]);
  s[1] = { "Alice", 2 , 43 };
  printStudent(s[1]);
  return 0;
```



## Explanation #

In the above program, we define a function printStudent on **Line No. 12** that takes a structure variable s in its arguments and performs an operation on it.

#### Return a structure from a function #

So far, we have not seen a way to return multiple variables of different data types from a function. By returning a structure from a function, we can return multiple variables of different data types.

## Example program #

Press the **RUN** button and see the output!

```
#include <iostream>

using namespace std;

// Student structure
struct Student {
    string name;
    int roll_number;
    int marks;
};

// function fillStudent
Student fillStudent(string name, int roll_number, int marks) {
    Student s;
    s.name = name;
    s.roll_number = roll_number;
}
```

```
s.marks = marks;
  return s;
// printStudent function prints the members of structure variable
void printStudent(struct Student s) {
  cout << "Student information" << endl;</pre>
  cout << "Name = " << s.name << endl;</pre>
  cout << "Roll Number = " << s.roll_number << endl;</pre>
  cout << "Marks = " << s.marks << endl;</pre>
}
int main() {
  struct Student s[100];
  s[0] = fillStudent("John", 1, 50);
  printStudent(s[0]);
  s[1] = fillStudent("Alice", 2, 43);
  printStudent(s[1]);
  return 0;
```







רח

## **Explanation** #

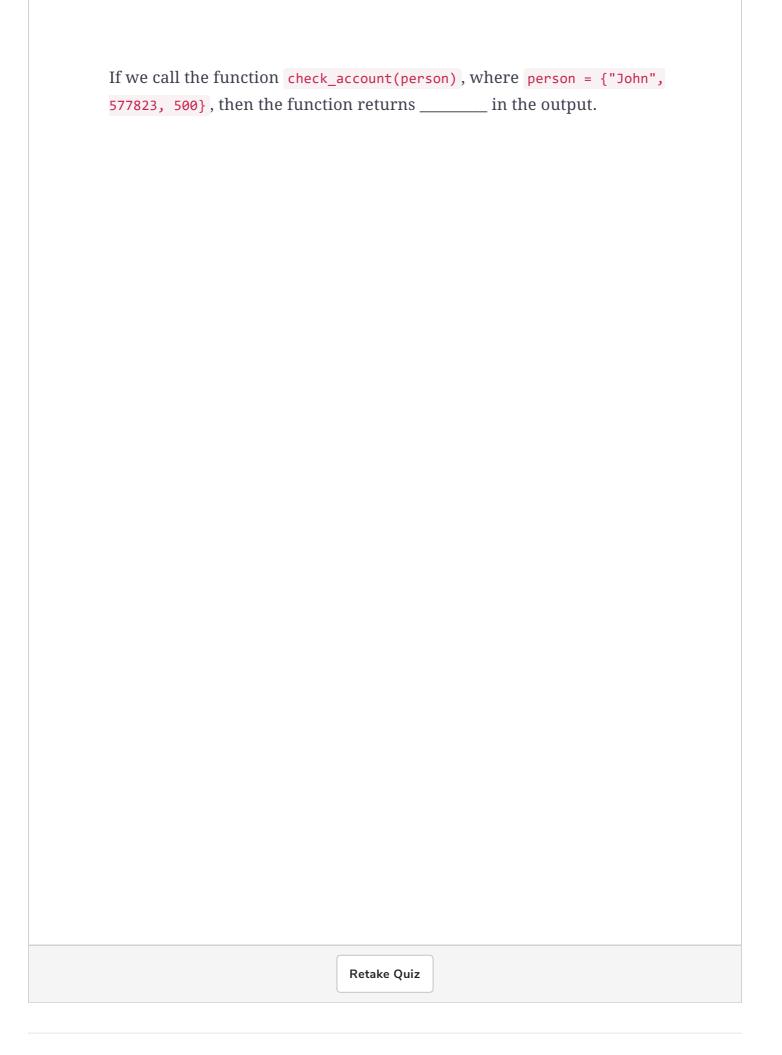
In the above program, we define a function fillStudent on Line No. 11 that takes one string value and the two int values in the input, performs an operation on it, and then returns the structure in the output.

Q !

Consider the code given below:

```
struct Account {
    string name;
    int number;
    double balance;
};

int check_account(struct Account p) {
    if (p.balance < 500) {
        return 0;
    } else {
        return -1;
    }
}</pre>
```



In the next lesson, you will learn how to declare a pointer to the structure.