Experiment 5

AIM: Write a program to swap 2 numbers both by call by value and call by reference, using 2 functions swap_value() and swap_reference(), respectively, by getting choice from the user and executing the user's choice by switch case.

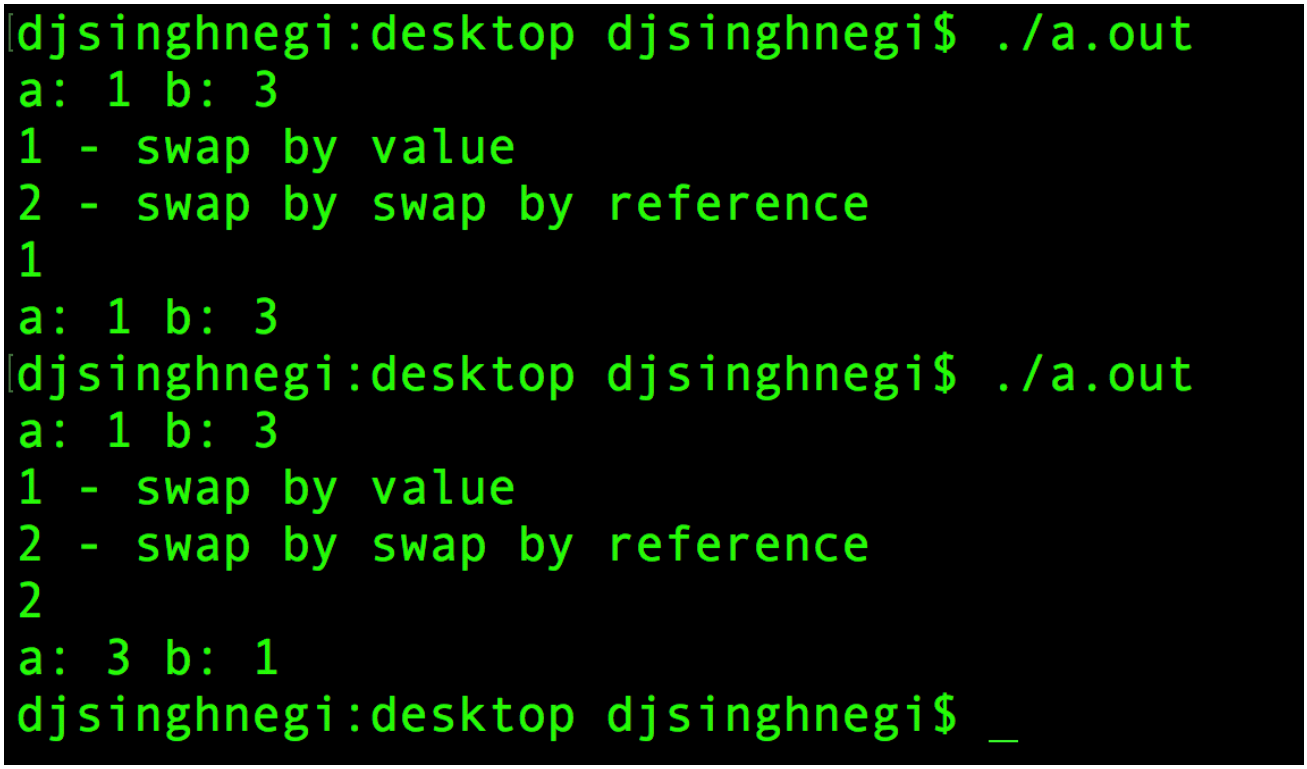
Theory:

1. The **call by reference** method of passing arguments to a function copies the reference of an argument into the formal parameter. Inside the function, the reference is used to access the actual argument used in the call. This means that changes made to the parameter affect the passed argument.
2. The **call by value** method of passing arguments to a function copies the actual value of an argument into the formal parameter of the function. In this case, changes made to the parameter inside the function have no effect on the argument

Code:

```
#include < iostream > using namespace std;
void swap_value(int a, int b) {
    int c = b;
    b = a;
    a = c;
}
void swap_reference(int & a, int & b) {
    int c = b;
    b = a;
    a = c;
}
int main() {
    int a = 1, b = 3;
    cout << "a: " << a << " b: " << b << endl;
    cout << "1 - swap by value\n";
    cout << "2 - swap by swap by reference\n";
    int choice;
    cin >> choice;
    switch (choice) {
        case 1:
            swap_value(a, b);
            cout << "a: " << a << " b: " << b << endl;
            break;
        case 2:
            swap_reference(a, b);
            cout << "a: " << a << " b: " << b << endl;
            break;
    }
    return 0;
}
```

Output:



Discussion:

The function swap_value swaps the the variables by value and hence no change is noticed in the global scope, whereas in swap_reference the address is passed and hence the values actually get swapped.