First,  the code is compiled successfully, and the result is not correct.

```
Enter the array size (>0) and the numbers to fill the array with: 5
3
1
4
2
6
=== Array before Sorting = 3, 1, 4, 2, 6
=== Array after Sorting = 32764, 32764, 32764, 32764, 32764
```

Then, log into gdb and set a break point. Run with the break point. It shows that every thing goes fine before sorting.

```
(gdb) file ArraySort
Reading symbols from /home1/ugrads/jmiao/CS2263_Summer2019_L2/ArraySort...(no de
bugging symbols found)...done.
(gdb) b swapAdjacent
Breakpoint 1 at 0x40068d
(gdb) r
Starting program: /home1/ugrads/jmiao/CS2263_Summer2019_L2/ArraySort
Enter the array size (>0) and the numbers to fill the array with: 5
3
1
4
2
6
=== Array before Sorting = 3, 1, 4, 2, 6

Breakpoint 1, 0x000000000040068d in swapAdjacent ()
Missing separate debuginfos, use: debuginfo-install glibc-2.17-222.el7.x86_64
```

Backtracing like this:

```
(gdb) bt
#0  0x000000000040068d in swapAdjacent ()
#1  0x0000000000400783 in inPlaceSort ()
#2  0x00000000004008ea in main ()
```

Then, correct the swapAdjacent and compare function.

```
Enter the array size (>0) and the numbers to fill the array with: 5
3
1
4
2
6
=== Array before Sorting = 3, 1, 4, 2, 6
=== Array after Sorting = 1, 2, 3, 4, 6
```

Then, I found the reason why I can’t pass the test. It looks a little bit different. So I delete two printf instructions in main function.

```
Enter the array size (>0) and the numbers to fill the array with: 5, 4, 3, 2, 1
1, 2, 3, 4, 5
```

Open | test1.result ~/CS2263_Summer2019_L2 | Save

```
Enter the array size (>0) and the numbers to fill the array with: === Array before Sorting = 5, 4,
3, 2, 1
=== Array after Sorting = 1, 2, 3, 4, 5
```

Finally, I passed the test!

```
(gdb) make test
gcc -Wall -Wextra -c ArraySort.c
gcc -Wall -Wextra -o ArraySort ArraySort.o
./ArraySort < Data/test1.input > test1.result
./TestPassed.sh test1.result Data/test1.expected

######   Passed   ###### test1.result is equal to Data/test1.expected

./ArraySort < Data/test2.input > test2.result
./TestPassed.sh test2.result Data/test2.expected

######   Passed   ###### test2.result is equal to Data/test2.expected

./ArraySort < Data/test3.input > test3.result
./TestPassed.sh test3.result Data/test3.expected

######   Passed   ###### test3.result is equal to Data/test3.expected
```