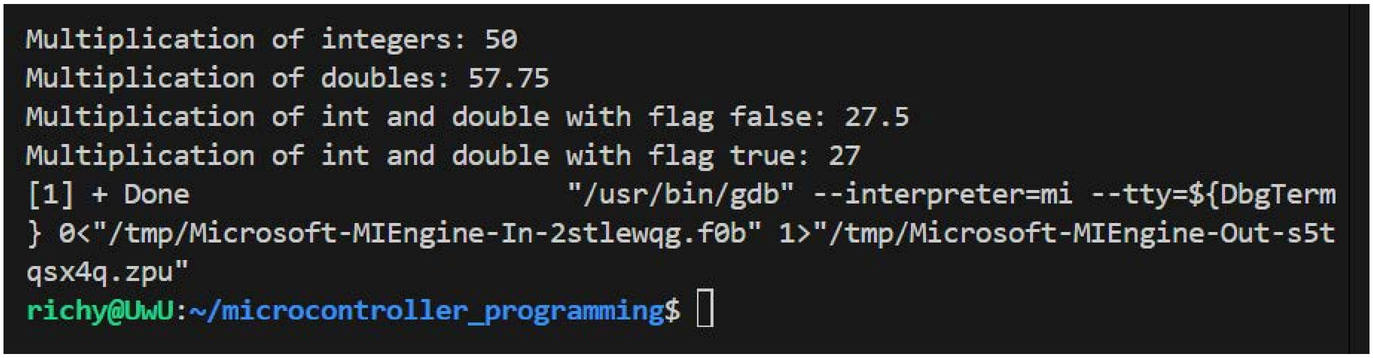
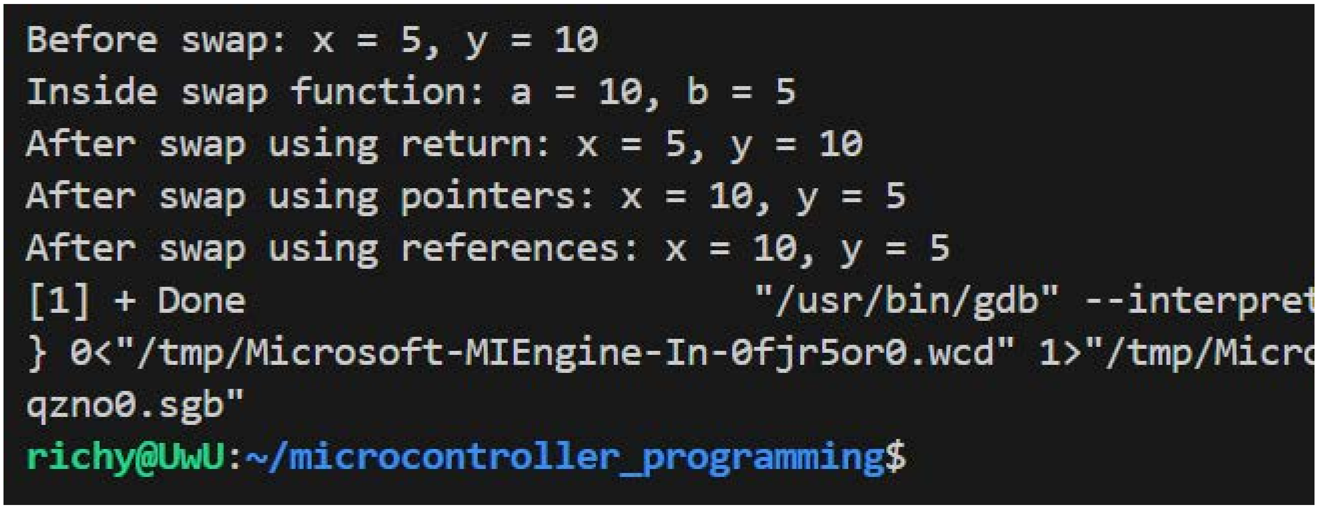
Task1



```
Multiplication of integers: 50
Multiplication of doubles: 57.75
Multiplication of int and double with flag false: 27.5
Multiplication of int and double with flag true: 27
[1] + Done                       "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm
} 0<"/tmp/Microsoft-MIEngine-In-2stlewqg.f0b" 1>"/tmp/Microsoft-MIEngine-Out-s5t
qsx4q.zpu"
richy@UwU:~/microcontroller_programming$ 
```

Task2

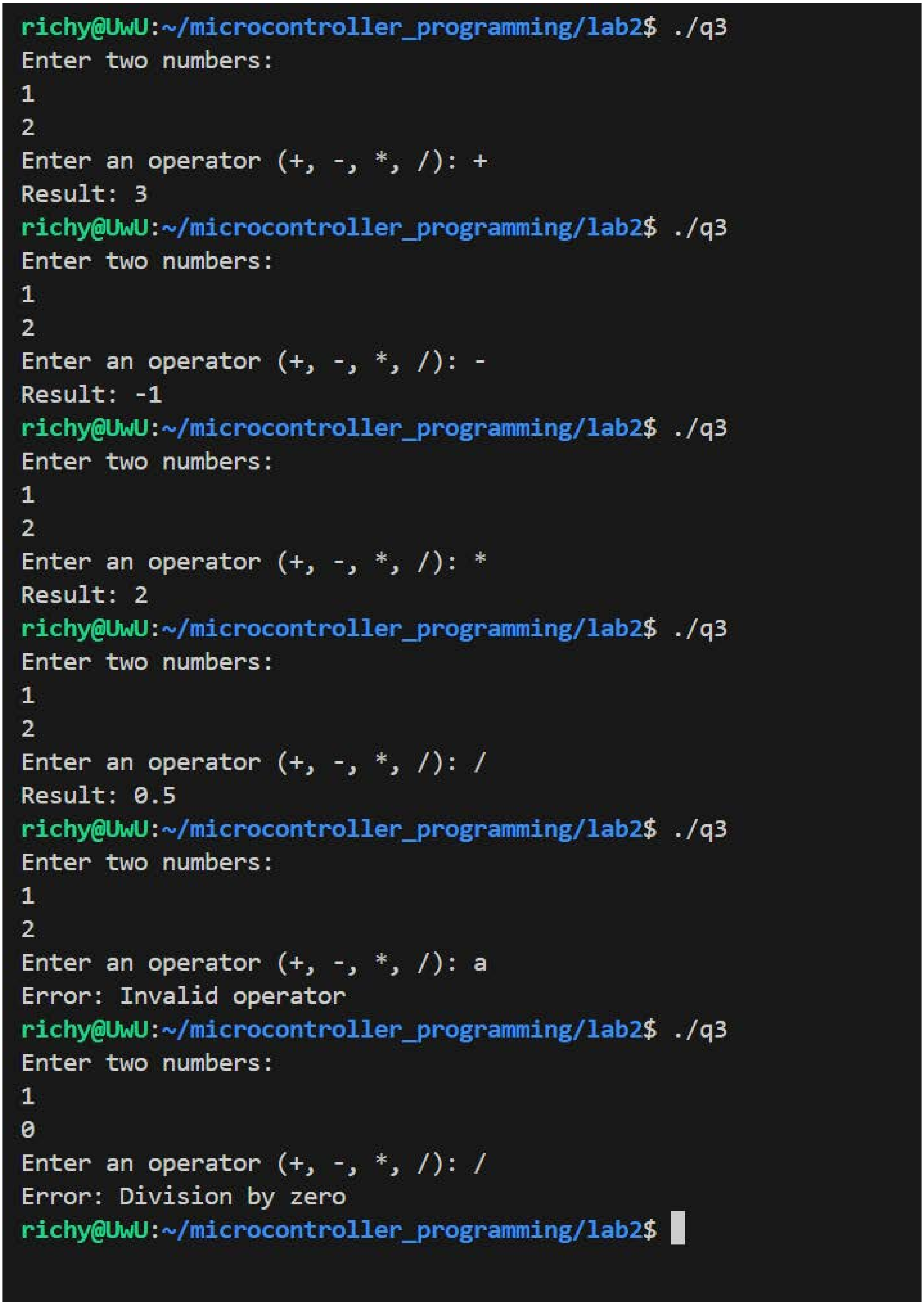


```
Before swap: x = 5, y = 10
Inside swap function: a = 10, b = 5
After swap using return: x = 5, y = 10
After swap using pointers: x = 10, y = 5
After swap using references: x = 10, y = 5
[1] + Done                       "/usr/bin/gdb" --interpret
} 0<"/tmp/Microsoft-MIEngine-In-0fjr5or0.wcd" 1>"/tmp/Micro
qzno0.sgb"
richy@UwU:~/microcontroller_programming$
```

Using pointer and reference could be used to swap the values of two variables
not using pointer and reference will not swap the values of two variables as C++ can only return one variable for a function

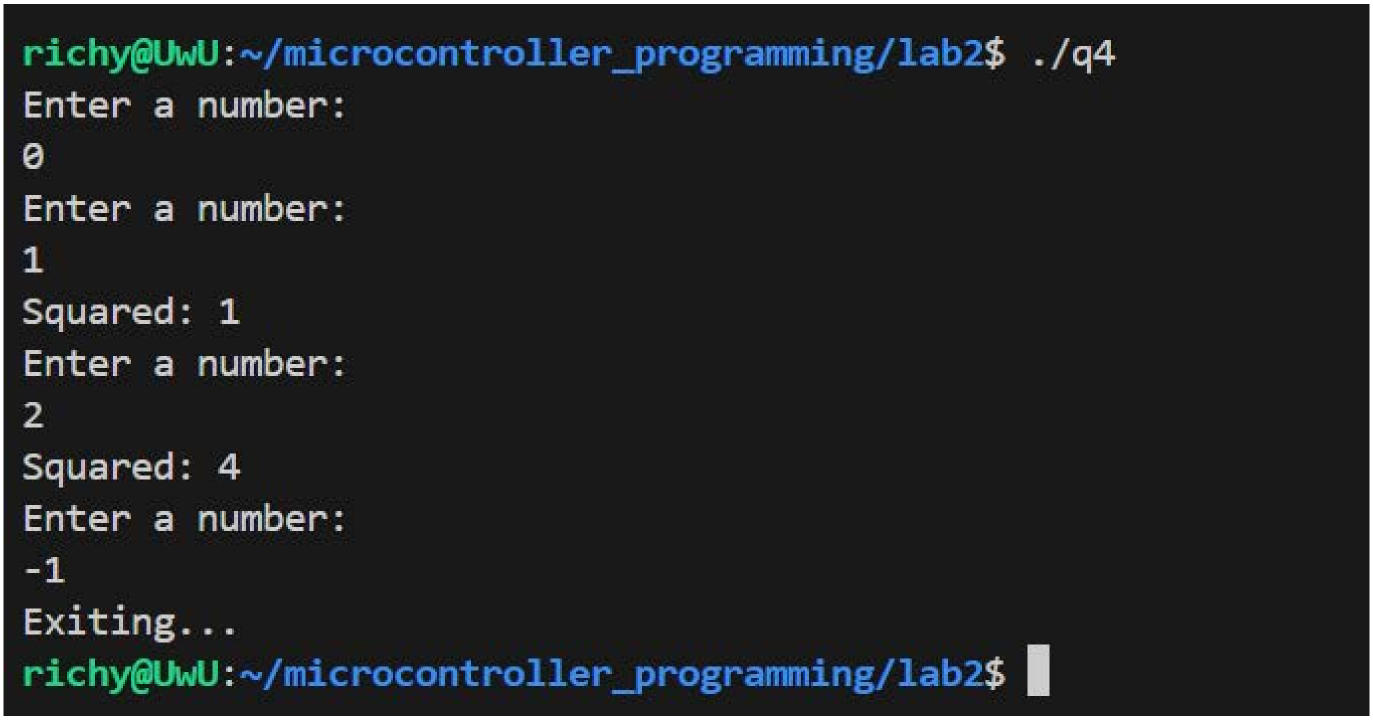
The best practice is to use reference as it is more readable and less error-prone

Task3



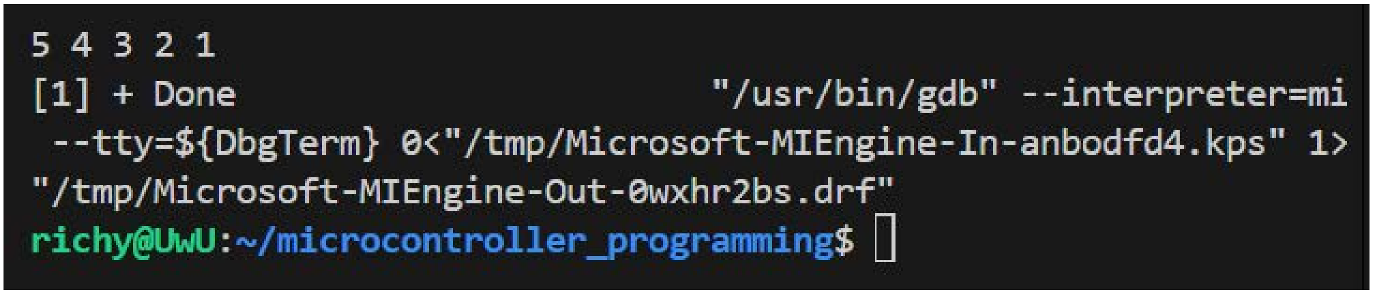
```
richy@UwU:~/microcontroller_programming/lab2$ ./q3
Enter two numbers:
1
2
Enter an operator (+, -, *, /): +
Result: 3
richy@UwU:~/microcontroller_programming/lab2$ ./q3
Enter two numbers:
1
2
Enter an operator (+, -, *, /): -
Result: -1
richy@UwU:~/microcontroller_programming/lab2$ ./q3
Enter two numbers:
1
2
Enter an operator (+, -, *, /): *
Result: 2
richy@UwU:~/microcontroller_programming/lab2$ ./q3
Enter two numbers:
1
2
Enter an operator (+, -, *, /): /
Result: 0.5
richy@UwU:~/microcontroller_programming/lab2$ ./q3
Enter two numbers:
1
2
Enter an operator (+, -, *, /): a
Error: Invalid operator
richy@UwU:~/microcontroller_programming/lab2$ ./q3
Enter two numbers:
1
0
Enter an operator (+, -, *, /): /
Error: Division by zero
richy@UwU:~/microcontroller_programming/lab2$
```

Task4



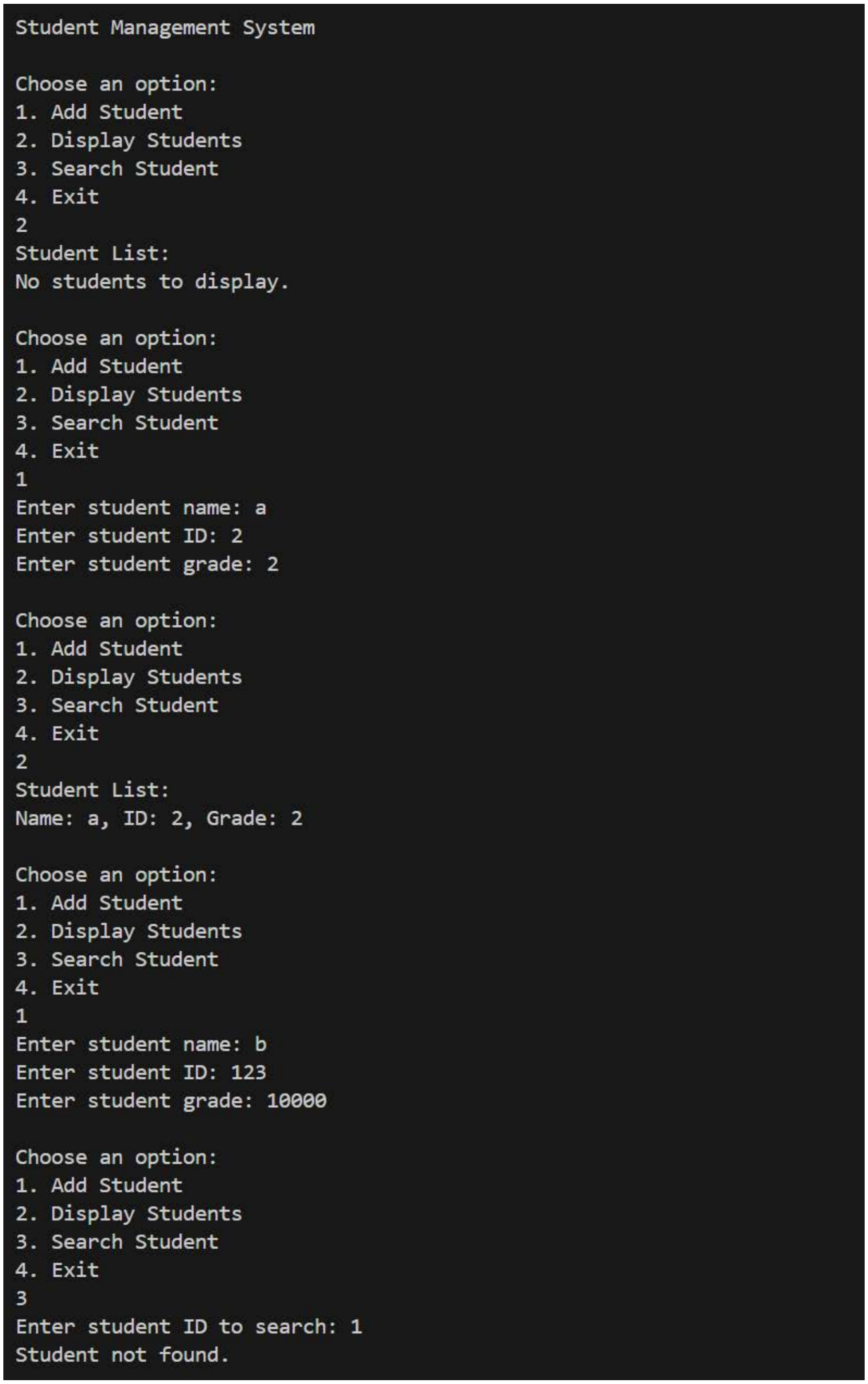
```
richy@UwU:~/microcontroller_programming/lab2$ ./q4
Enter a number:
0
Enter a number:
1
Squared: 1
Enter a number:
2
Squared: 4
Enter a number:
-1
Exiting...
richy@UwU:~/microcontroller_programming/lab2$ 
```

Task5

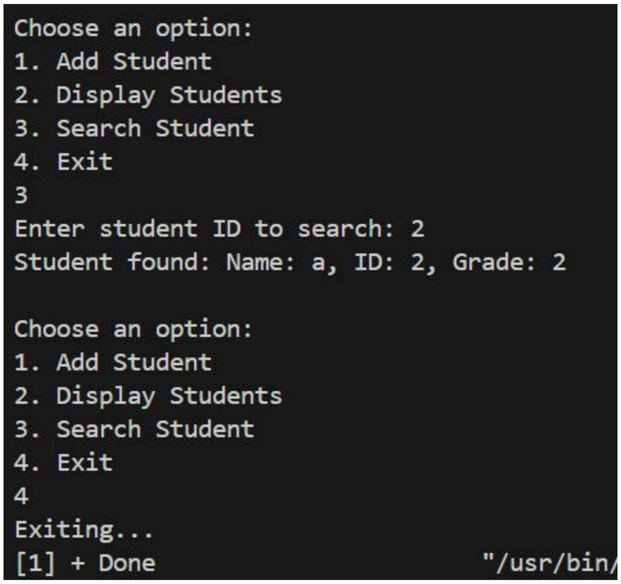


```
5 4 3 2 1
[1] + Done                       "/usr/bin/gdb" --interpreter=mi
 --tty=${DbgTerm} 0<"/tmp/Microsoft-MIEngine-In-anbodfd4.kps" 1>
"/tmp/Microsoft-MIEngine-Out-0wxhr2bs.drf"
richy@UwU:~/microcontroller_programming$ 
```

Task6

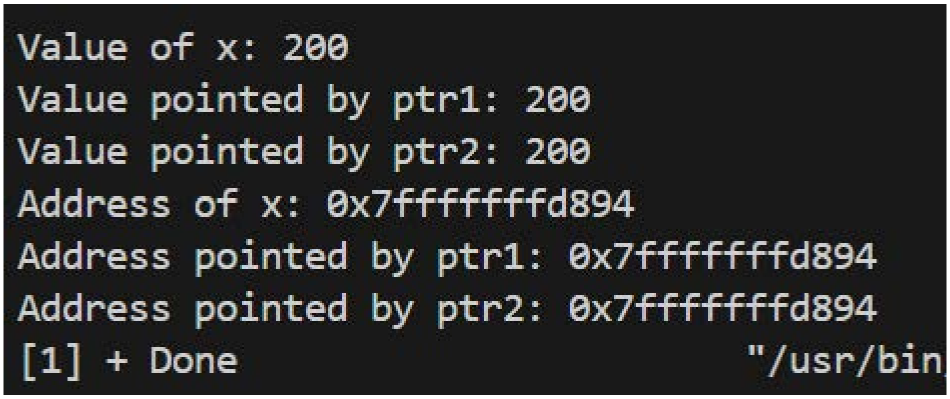


```
Student Management System

Choose an option:
1. Add Student
2. Display Students
3. Search Student
4. Exit
2
Student List:
No students to display.

Choose an option:
1. Add Student
2. Display Students
3. Search Student
4. Exit
1
Enter student name: a
Enter student ID: 2
Enter student grade: 2

Choose an option:
1. Add Student
2. Display Students
3. Search Student
4. Exit
2
Student List:
Name: a, ID: 2, Grade: 2

Choose an option:
1. Add Student
2. Display Students
3. Search Student
4. Exit
1
Enter student name: b
Enter student ID: 123
Enter student grade: 10000

Choose an option:
1. Add Student
2. Display Students
3. Search Student
4. Exit
3
Enter student ID to search: 1
Student not found.
```



```
Choose an option:
1. Add Student
2. Display Students
3. Search Student
4. Exit
3
Enter student ID to search: 2
Student found: Name: a, ID: 2, Grade: 2

Choose an option:
1. Add Student
2. Display Students
3. Search Student
4. Exit
4
Exiting...
[1] + Done                     "/usr/bin/
```

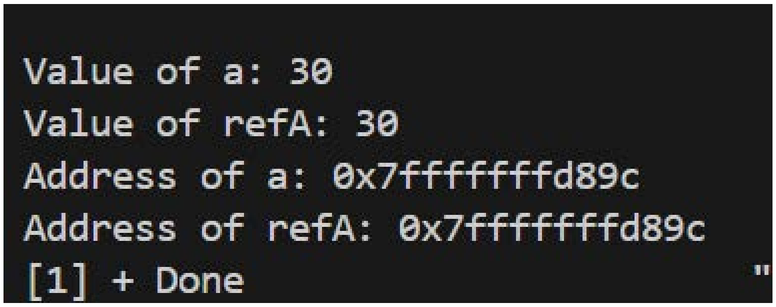
Task7



```
Value of x: 200
Value pointed by ptr1: 200
Value pointed by ptr2: 200
Address of x: 0x7ffffffd894
Address pointed by ptr1: 0x7ffffffd894
Address pointed by ptr2: 0x7ffffffd894
[1] + Done                          "/usr/bin
```

With shallow copy, the object pointed by other pointer would also change if the object of copied pointer(i.e. x) is changed

Task8



```

Value of a: 30
Value of refA: 30
Address of a: 0x7ffffffd89c
Address of refA: 0x7ffffffd89c
[1] + Done                          "
```

variable a would also change accordingly if refA is changed