Comsats University Vehari Campus

DS Lab Assignment 01

Submitted by:

Mubeshra Shafique

Roll No.:

Sp22-BCS-120

Section:

B

Subject:

Data Structure

Submitted to:

Mam Yasmeen Jana

Pointers Program

Program 01:

```
#include <iostream>
using namespace std;
int main() {
    int* ptr = new int; // Allocate memory for an integer
dynamically
    *ptr = 10;
    cout << "Value stored in dynamically allocated memory: " <<
*ptr << endl;
    delete ptr; // Deallocate memory to prevent memory leaks
    return 0;
}
```

Output:

Compile Result

```
Value stored in dynamically allocated me
mory: 10

[Process completed - press Enter]
```

Program 02:

```
int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    int* ptr = arr;

cout << "Array elements using pointers: ";
for (int i = 0; i < 5; ++i) {
    cout << *ptr << " ";
    ptr++;
}
```

```
    cout << endl;
    return 0;
}
```

Output:

Compile Result

```
Array elements using pointers: 1 2 3 4 5

[Process completed - press Enter]
```

Program 03:

```
#include <iostream>

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

int main() {
    int num1 = 5, num2 = 10;

    std::cout << "Before swapping: num1 = " << num1 << ",
num2 = " << num2 << std::endl;

    swap(&num1, &num2);

    std::cout << "After swapping: num1 = " << num1 << ", num2
= " << num2 << std::endl;
```

```
    return 0;
}
```

Output:

```
Compile Result

Maximum element in the array: 20

[Process completed - press Enter]
```

Program 04:

```
#include <iostream>
using namespace std;

int main() {
    int arr[] = {10, 20, 30, 40, 50};
    int* ptr = arr;

    cout << "Array elements using pointer arithmetic: ";
    for (int i = 0; i < 5; ++i) {
        cout << *ptr << " ";
        ptr++; // Move the pointer to the next element
    }
    cout << endl;

    return 0;
}
```

Output:

```
Compile Result

Array elements using pointer arithmetic:
 10 20 30 40 50

[Process completed - press Enter]
```

Program 05:

```
#include <iostream>

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
int main() {
    int num1 = 5, num2 = 10;

    std::cout << "Before swapping: num1 = " << num1 << ",
num2 = " << num2 << std::endl;

    swap(&num1, &num2);

    std::cout << "After swapping: num1 = " << num1 << ", num2
= " << num2 << std::endl;

    return 0;
}
```

Output:

Compile Result

```
Before swapping: num1 = 5, num2 = 10
After swapping: num1 = 10, num2 = 5

[Process completed - press Enter]
```

Program 06:

```
#include <iostream>
using namespace std;

void showMessage() {
    cout << "Hello, World!" << endl;
}
int main() {
    void (*ptr)() = showMessage; // Pointer to a function
    ptr(); // Call the function using the pointer
   return 0;
}
```

Output:

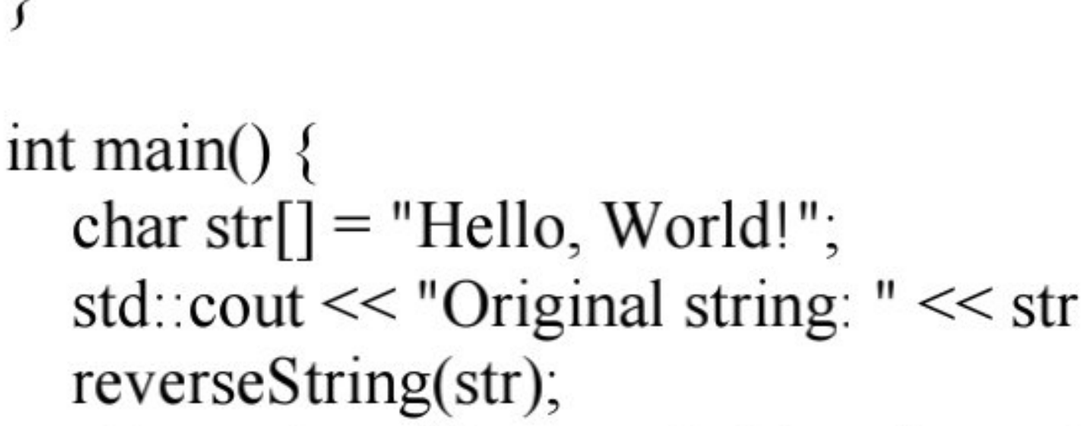
Compile Result

```
Hello, World!

[Process completed - press Enter]
```

Program 07:

```
include <iostream>
#include <cstring>

void reverseString(char *str) {
int len = strlen(str);
char *start = str;
char *end = str + len - 1;

while (start < end) {
    char temp = *start;
    *start = *end;
    *end = temp;
    start++;
    end--;
}
}

int main() {
    char str[] = "Hello, World!";
    std::cout << "Original string: " << str << std::endl;
    reverseString(str);
    std::cout << "Reversed string: " << str << std::endl;

    return 0;
    }
```

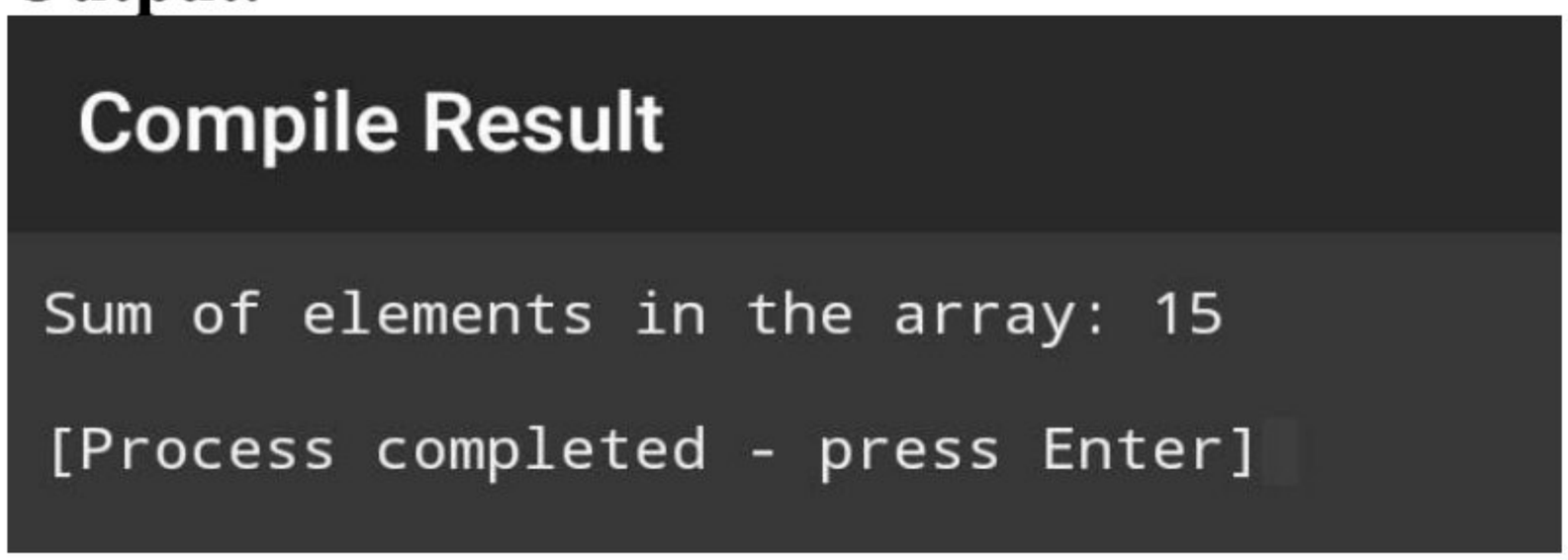
Output:



Program 08:

```
#include <stdio.h>

int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int *ptr = arr;
    int sum = 0;

    for (int i = 0; i < 5; i++) {
        sum += *ptr;
        ptr++;
    }

    printf("Sum of elements in the array: %d\n", sum);

    return 0;
}
```

Output:



Program 09:

```
#include <stdio.h>
#include <stdlib.h>

int main() {
    int *arr;
    int n;
```

```
printf("Enter the number of elements: ");
scanf("%d", &n);

arr = (int *)malloc(n * sizeof(int));

if (arr == NULL) {
    printf("Memory allocation failed\n");
    return 1;
}
printf("Enter %d elements:\n", n);
for (int i = 0; i < n; i++) {
    scanf("%d", &arr[i]);
}
printf("Elements entered by the user: ");
for (int i = 0; i < n; i++) {
    printf("%d ", arr[i]);
}
free(arr);
return 0;}
```

Output:

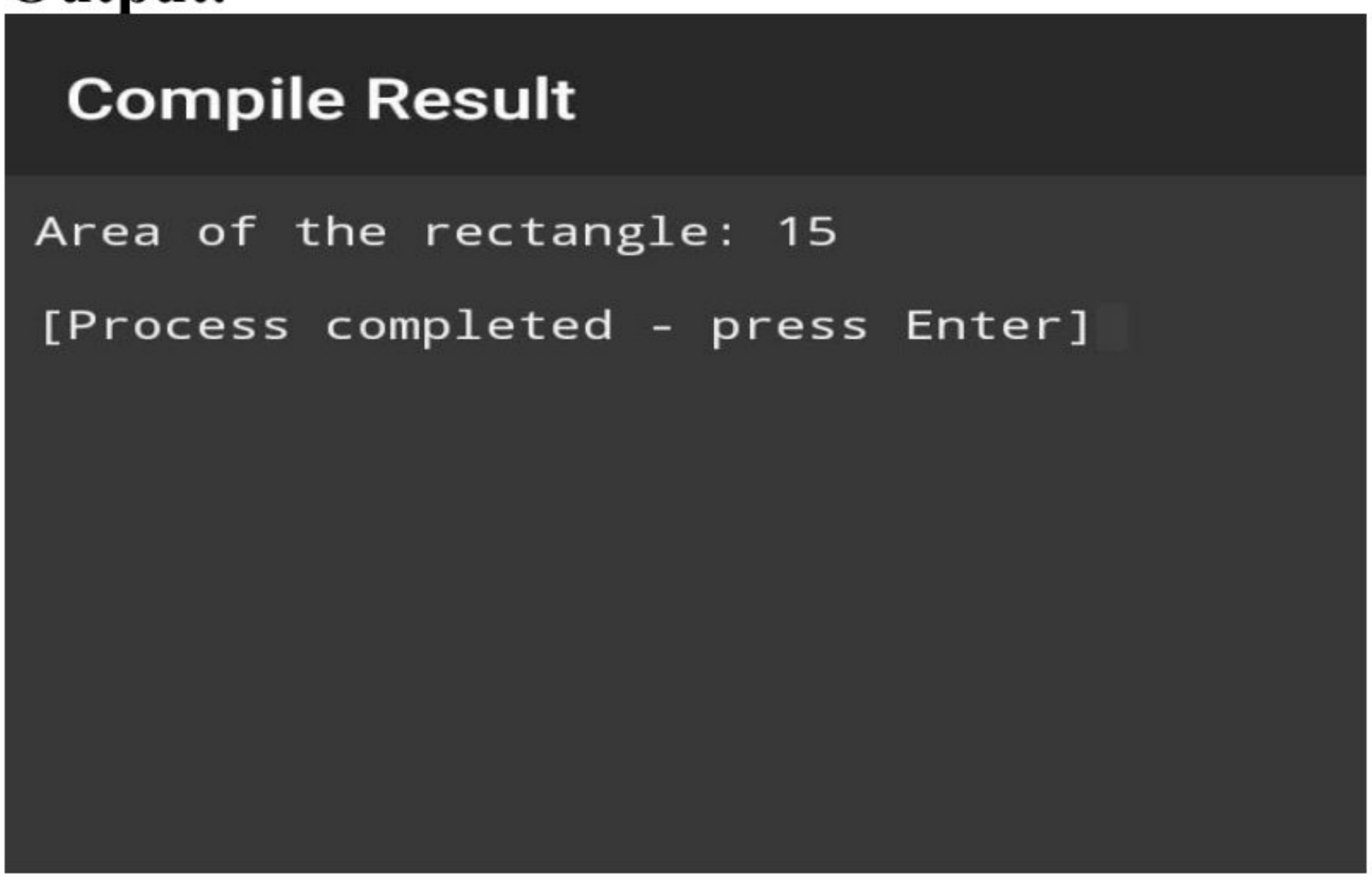
Compile Result

```
Enter the number of elements: 3
Enter 3 elements:
4
5
6
Elements entered by the user: 4 5 6
[Process completed - press Enter]
```

Program 010:

```
#include <iostream>
class Rectangle {
public:
    int length;
    int width;
    Rectangle(int l, int w) : length(l), width(w) {}

    int area() {
        return length * width;
    }
};
int main() {
    Rectangle r(5, 3);
    Rectangle *ptr = &r;
    std::cout << "Area of the rectangle: " << ptr->area() <<
std::endl;
    return 0;
}
```

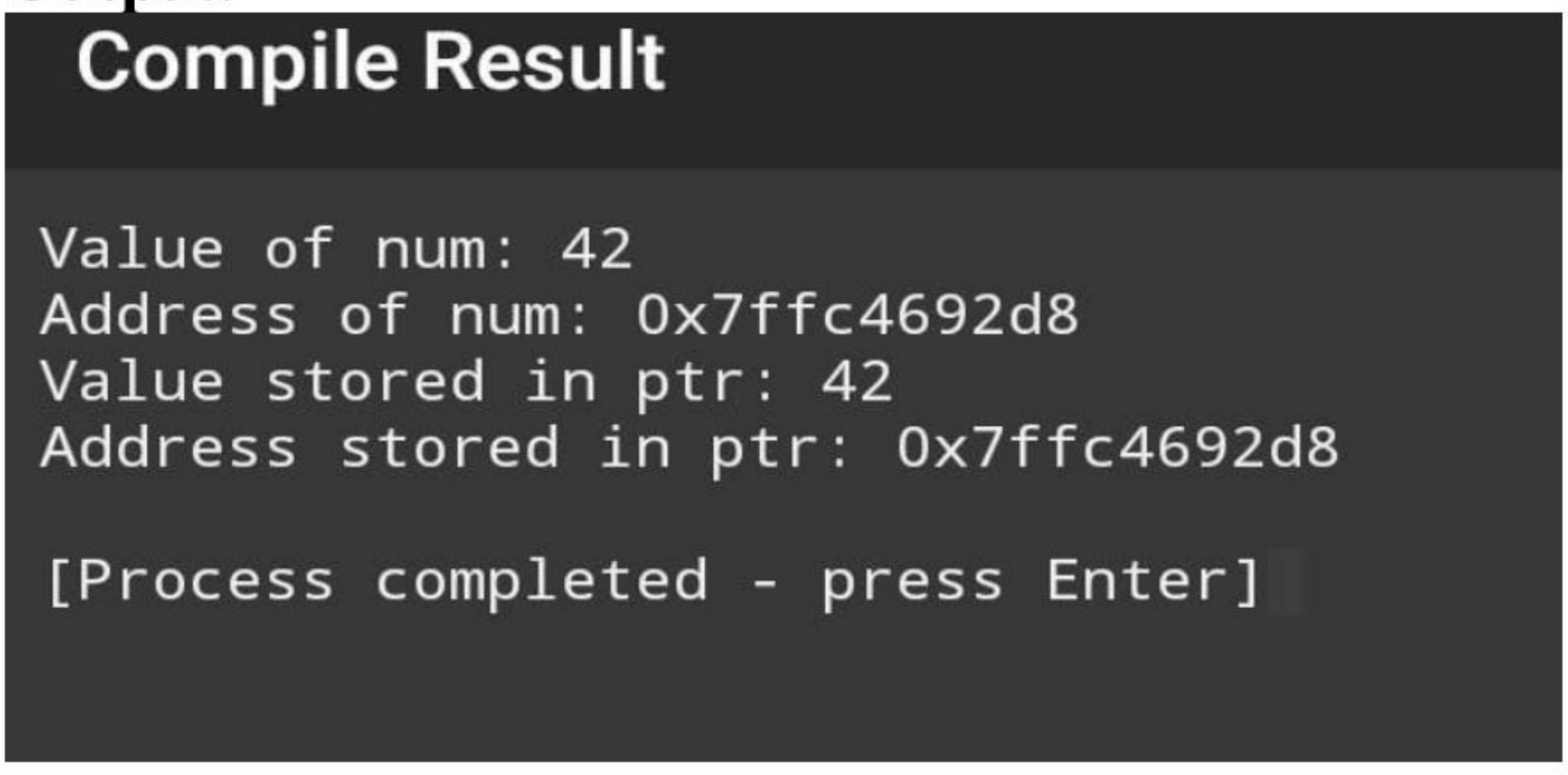
Output:



Program 011:

```
#include <iostream>
using namespace std;

int main() {
    int num = 42;
    int* ptr = &num;

    cout << "Value of num: " << num << endl;
    cout << "Address of num: " << &num << endl;
    cout << "Value stored in ptr: " << *ptr << endl;
    cout << "Address stored in ptr: " << ptr << endl;

    return 0;
}
```

Output:



Program 012:

```
#include <iostream>

int add(int a, int b) {
    return a + b;
}
int subtract(int a, int b) {
    return a - b;
}
int main() {
    int (*ptr)(int, int);
    ptr = add;
    int result = ptr(5, 3);
    std::cout << "Result of addition: " << result << std::endl;
    ptr = subtract;
    result = ptr(5, 3);
    std::cout << "Result of subtraction: " << result << std::endl;
    return 0;
}
```

Output:

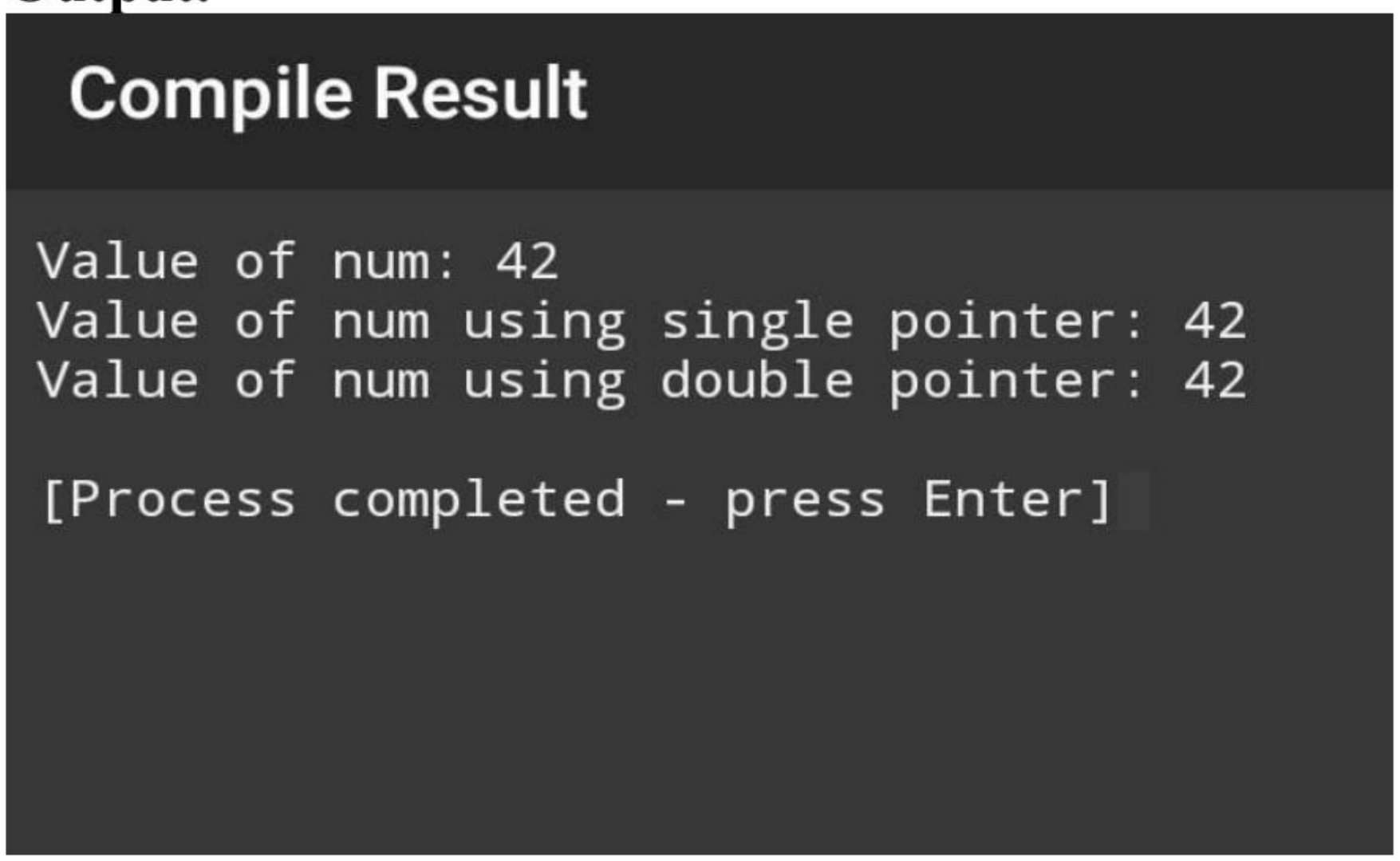
Compile Result

```
Result of addition: 8
Result of subtraction: 2

[Process completed - press Enter]
```

Program 013:

```
#include <iostream>

int main() {
    int num = 42;
    int *ptr = &num;
    int **ptr2 = &ptr;

    std::cout << "Value of num: " << num << std::endl;
    std::cout << "Value of num using single pointer: " << *ptr <<
std::endl;
    std::cout << "Value of num using double pointer: " << **ptr2
<< std::endl;

    return 0;
}
```

Output:



Program 14:

```
#include <iostream>

void modifyValue(int *x) {
    (*x) += 5;
}

int main() {
    int num = 10;

    std::cout << "Original value of num: " << num << std::endl;
    modifyValue(&num);
    std::cout << "Modified value of num: " << num << std::endl;

    return 0;
}
```

Output:

Compile Result

```
Original value of num: 10
Modified value of num: 15

[Process completed - press Enter]
```

Program 15:

```
#include <iostream>

int main() {
    const int num = 42;
    const int *ptr = &num; // Pointer to constant data

    std::cout << "Value through ptr: " << *ptr << std::endl;

    // Attempting to modify the value through the pointer will result in an error:
    // *ptr = 50; // Error

    return 0;
}
```

Output: