COMSATS University Islamabad, Vehari Campus

Department of Computer Science

Class: BCS-SP22-4A

sp22-bcs-001

Ahmed Jamshed **Submission Deadline: 10 Sep 2023**

Subject: Data Structures and Algorithms-Lab **Instructor: Yasmeen Jana**

Max Marks: 10 **Reg. No:**

-

Email: yasmeenjana@cuivehari.edu.pk

You can ask queries related to Lab Activities on the above email.

Activity 1:

Create a GitHub Account. Make a repository with the name "**DSA_Lab". Mention the link here after the account creation.**

Solution:

https://github.com/Ahmedjamshed7/DSA_lab.git

Activity 2:

Write any 15 programs that will explain the concepts of pointers.

In this file, you should place the code and its output screenshot.

After completing the activities, Upload the final pdf and code to the "**DSA_Lab"** repository.

Pointer Declaration and Initialization:

```
#include <iostream>

int main() {
    int *ptr;
    int number = 10;
    ptr = &number;

    std::cout << "Value of number: " << *ptr << std::endl;

    return 0;
}
```

Pointer Arithmetic (Increment):

```
#include <iostream>

int main() {
    int array[] = {1, 2, 3, 4};
    int *ptr = array;
    ptr++;

    std::cout << "Value after increment: " << *ptr << std::endl;

    return 0;
}
```

Pointer Arithmetic (Decrement):

```
#include <iostream>

int main() {
    int array[] = {1, 2, 3, 4};
    int *ptr = array + 3;
    ptr--;

    std::cout << "Value after decrement: " << *ptr << std::endl;

    return 0;
}
```

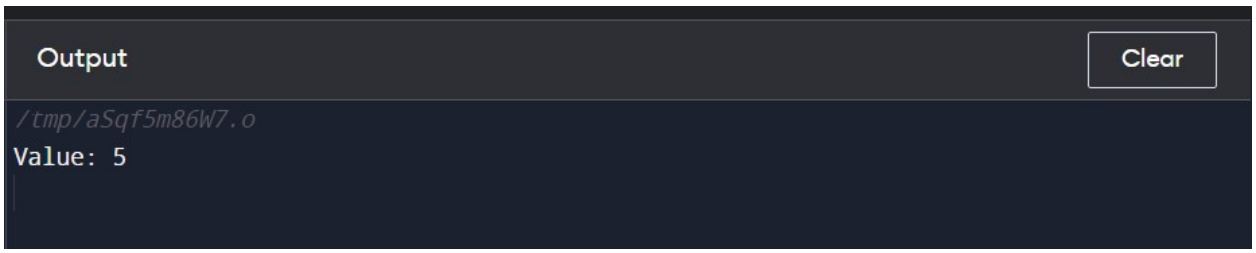
Pointer to Constant Value:

```
#include <iostream>

int main() {
    const int value = 5;
    const int *ptr = &value;

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

    return 0;
}
```



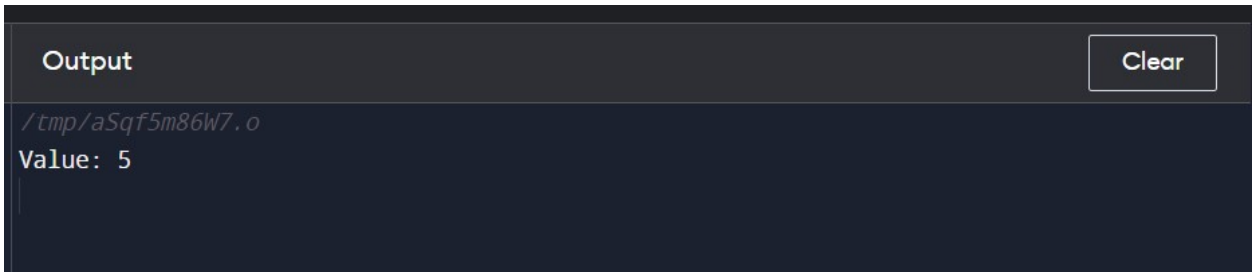
Pointer to Constant Value:

```
#include <iostream>

int main() {
    const int value = 5;
    const int *ptr = &value;

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

    return 0;
}
```



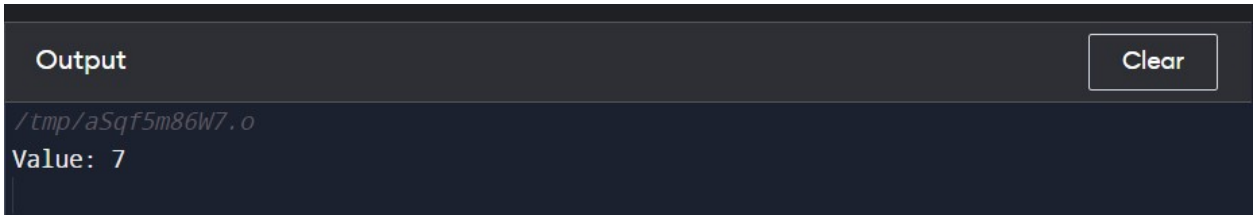
Constant Pointer:

```
#include <iostream>

int main() {
    int number = 7;
    int *const ptr = &number;

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

    return 0;
}
```



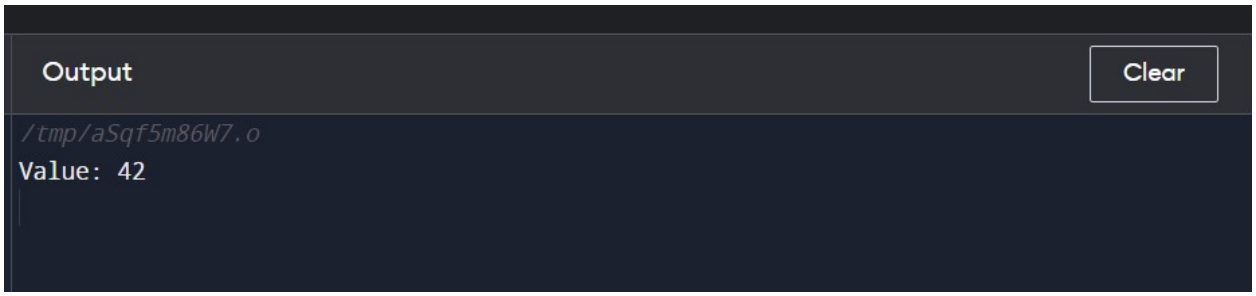
Pointer to Pointer:

```
#include <iostream>

int main() {
    int num = 42;
    int *ptr1 = &num;
    int **ptr2 = &ptr1;

    std::cout << "Value: " << **ptr2 << std::endl;

    return 0;
}
```



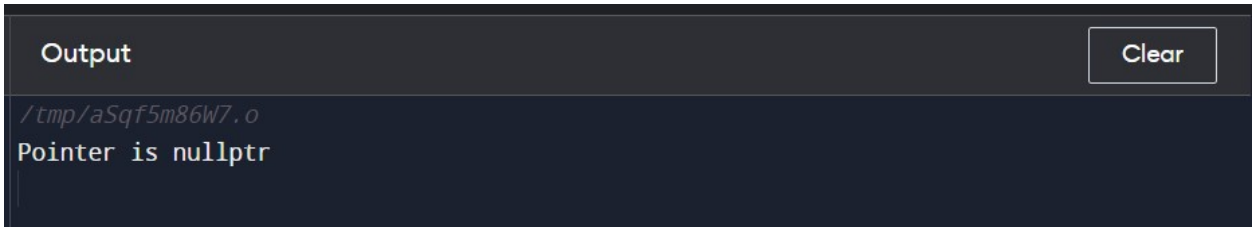
Null Pointer:

```
#include <iostream>

int main() {
    int *ptr = nullptr;

    if (ptr == nullptr) {
        std::cout << "Pointer is nullptr" << std::endl;
    }

    return 0;
}
```



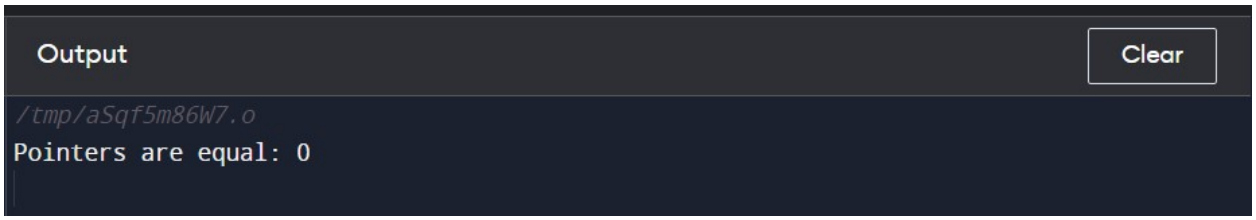
Pointer Comparison:

```
#include <iostream>

int main() {
    int a = 10, b = 20;
    int *ptr1 = &a, *ptr2 = &b;
    bool isEqual = (ptr1 == ptr2);

    std::cout << "Pointers are equal: " << isEqual << std::endl;

    return 0;
}
```



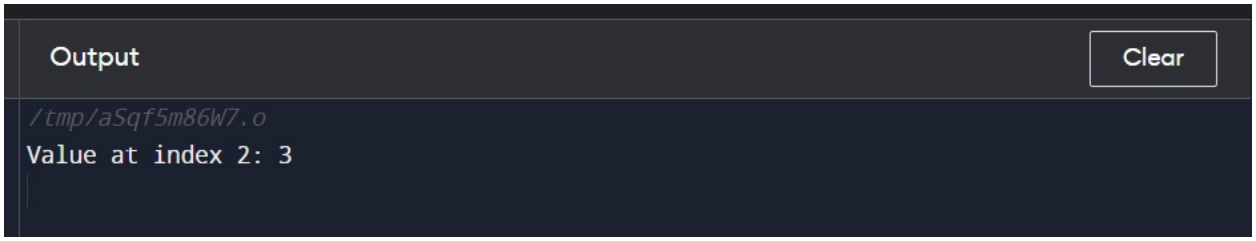
Pointer to Array:

```
#include <iostream>

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

    std::cout << "Value at index 2: " << ptr[2] << std::endl;

    return 0;
}
```



Pointer Arithmetic (Array Access):

```
#include <iostream>

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

    std::cout << "Third element: " << thirdElement << std::endl;

    return 0;
}
```