Activity No. n Pointers	
Replace with Title	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed: 09/18/25**
Section: CPE11S1	**Date Submitted:09/25/25**
Name(s): Cenndy M. Nieles	**Instructor: Engr. Jimlord M. Quejado**
6. Output	

1.What is a pointer in C++?

In C++ the pointers is able that stores the memory address of another variable. Instead of holding a direct value like an integer or a character, a pointer holds the location in memory where another piece of data is stored.

2. How does a pointer differ from a regular variable?

A regular variable stores a data value (like a number or text), while a pointer is a special type of variable that stores the memory address of another variable. The core difference is what they "hold":

3. What operator is used to get the address of a variable?

This unary operator takes a variable as an operand and returns the memory address of that variable.

4.What operator is used to access the value stored at a pointer's address?

The operator used to access the value stored at a pointer's address represented by the asterisk symbol (`*`). This operator is also known as the indirection operator.When placed before a pointer variable (`*ptr`), it retrieves the actual data stored at the memory location that the pointer is pointing to.

5.Why are pointers important in C++? Give two uses.

Pointers are also values but are different from those we've operated with so far. The types which we've used are closely linked to computer data processing but have reflected our ideas and our intuition. All of us use integers in everyday life to count all countable things.

Address Operator & determines the address of any variable in C++. This address can be assigned to the pointer variable to initialize it.

Storing Address The process of accessing the value present at the memory address pointed by the pointer is called dereferencing. This is done with the help of dereferencing operator as shown:

7. Supplementary Activity

Identify the Output

For each code snippet, predict the output without compiling:

1.
```
#include <iostream>

int main() {
int *ptr=x;
std::cout<<*ptr;
return 0;
}

OUTPUT: 42
```

EXPLANATION: #include<iostream> the produced an array with four integer elements called arr. Contiguous memory locations are used to store the elements. Declare and initialize a size 4 integer array. initialize the pointer of 'p' to point to the first element of 'arr', then print the value it points to, which is now the second element. Return 0; means your input is finished then you are going to predict the output without using a compiler.

2.
```
#include<iostream>
int main() {
int a=5;
int b=10;
int*p=&b;
std::cout<<*p;
return 0;
}

OUTPUT: 10
```

EXPLANATION: The program can manage input and output tasks, such as printing to the console, thanks to the iostream library, which is included in this line. int main () { main function where the program's execution begins. int a=5 declares an integer variable and initializes it with the value is 5. * used to access the data at the memory address stored.Prints the value that the pointer p is pointing. Return 0; indicates that the program has finished successfully.

3.
```
#include<iostream>
int main() {
```

```
int arr[3]={10,20,30};
int*p=arr;
std::cout<<*p;
```

OUTPUT: 3

EXPLANATION:

4.
```
#include <iostream>

int main() {
int arr[4] = {2, 4, 6, 8};
int *p= arr;
p++;
std::cout << *p;
return 0;
}
```

OUTPUT: 4

EXPLANATION:

5.
```
#include<iostream>

int main() {
int arr[3] = {5,15,25};
int *p = arr;
std::cout<<*  (p + 2 );
return 0;
}
```

`OUTPUT: 3`

EXPLANATION:

Error Spotting

Identify and fix the error(if any) in the codes below.

1.
```
int arr[3] = {1, 2, 3};
      int *p = &arr;
```

Error: int *p = &arr; The error is that you're trying to assign a pointer to the entire array (`&arr`) to a pointer.

Correct code:

```
        #include<iostream>

        int main() {
        int arr[3] = {1,2,3};
        int *p = arr;
        std::cout<< *p;
        return 0;
        }
```
OUTPUT: 3

EXPLANATION: The name of the array (arr). An array's name in C++ serves as a pointer to its first element. When you use arr, its type automatically changes to int *, which appropriately matches the type of p. This is referred to as "array decay."

```
 2, int arr[5];
      int *p;
     p = arr[2];
```

Error: The error is that you are trying to assign a value arr[2] to a pointer variable p, which requires a memory address.

Correct code:
```
             #include<iostream>

              int main() {
              int arr[5] = {10, 20, 30, 40, 50};
              int *p;
              p = arr;
              std::cout << *p;
              return 0;
              }
```

EXPLANATION: An integer value is attempted to be entered into a variable intended to store address by the program.

```
  3. int arr[4] = {10, 20, 30, 40};
     cout << *arr[2];
```

Error: arr[2] is a Value: arr[2] retrieves the array's third element, which is 30. This is not a memory address; rather, it is a simple integer value.* is for Pointers: To retrieve the value stored at a particular memory address, use the dereference operator (*. Only pointer variables can use it.

Correct code:

```
        #include <iostream>

        int main() {
        int arr[4] = {10, 20, 30, 40};
        std::cout << arr[2];
        return 0;
        }
```

EXPLANATION: Since arr[2] is a value rather than a pointer, the code cout \< *arr[2]; is erroneous. The dereference operator * cannot be applied to a value. Since are using the array element's index to access it directly, remove the dereference operator *.

8. Conclusion

Arrays store multiple elements of the same data type in contiguous memory.Every element in an array is stored in contiguous memory, and each element has its own distinct address,Pointers in C++ are a fundamental concept, serving as variables that store the memory addresses of other variables. Instead of holding a direct value, a pointer holds the location in memory where a value is stored.allow programs to access and manipulate data in memory efficiently, making them a key feature for system-level programming and dynamic memory management. When we access a pointer directly, we get the address it holds, not the actual data stored at that location.The two parts of pointers is:
Address Operator & determines the address of any variable in C++. This address can be assigned to the pointer variable to initialize it.

Storing Address The process of accessing the value present at the memory address pointed by the pointer is called dereferencing. This is done with the help of dereferencing operator as shown: