Assignment No. 4.3

Pointers

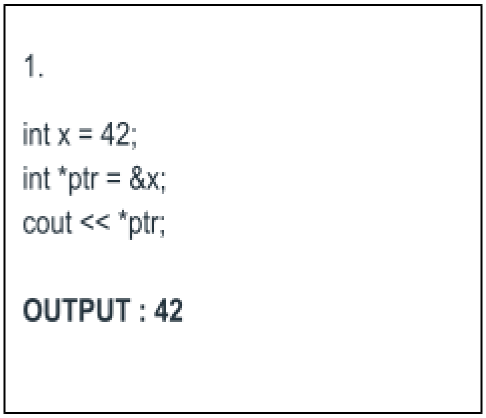
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed:** September 20, 2025
Section: CPE11S1	**Date Submitted:** September 22, 2025
Name(s): Mendoza, Nathaniel B.	**Instructor: Engr. Jimlord M. Quejado**

6. Output

1. **What is a pointer in C++?**
   A pointer is a variable that stores the memory address of another variable. So, it holds the location in memory where a specific value is stored.
2. **How does a pointer differ from a regular variable?**
   A regular variable stores a value, while a pointer stores the address of a value.
3. **What operator is used to get the address of a variable?**
   The ampersand (&) operator.
4. **What operator is used to access the value stored at a pointer's address?**
   The asterisk (*) operator (dereference operator).
5. **Why are pointers important in C++? Give two uses.**
   Because it enables efficient array and function handling and it allows dynamic memory allocation.

7. Supplementary Activity

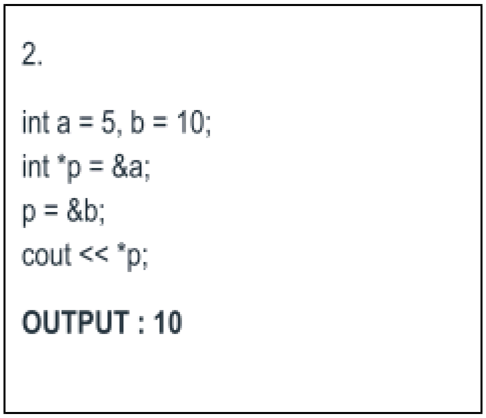
Predicting the OUTPUT



1.

```
int x = 42;
int *ptr = &x;
cout << *ptr;
```

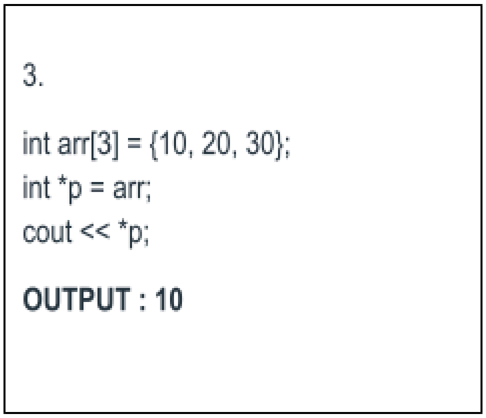
OUTPUT : 42



2.

```
int a = 5, b = 10;
int *p = &a;
p = &b;
cout << *p;
```

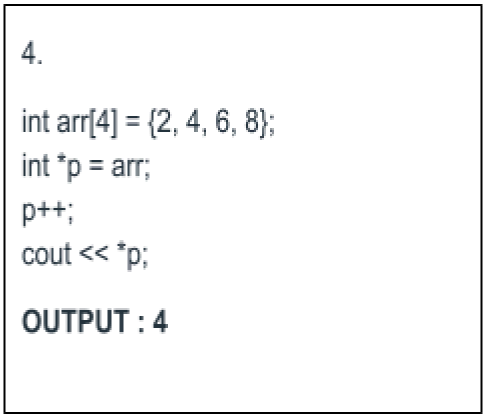
OUTPUT : 10



3.

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

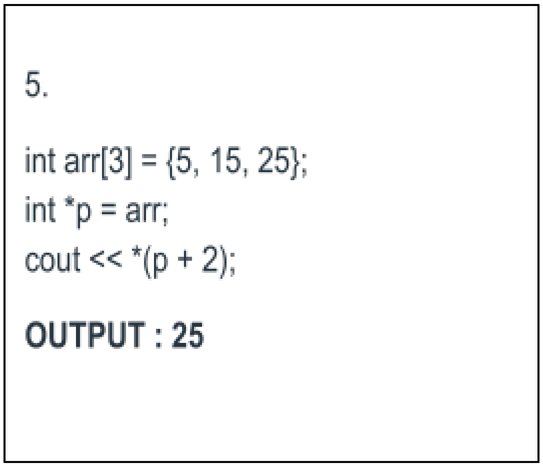
OUTPUT : 10



4.

```
int arr[4] = {2, 4, 6, 8};
int *p = arr;
p++;
cout << *p;
```

OUTPUT : 4

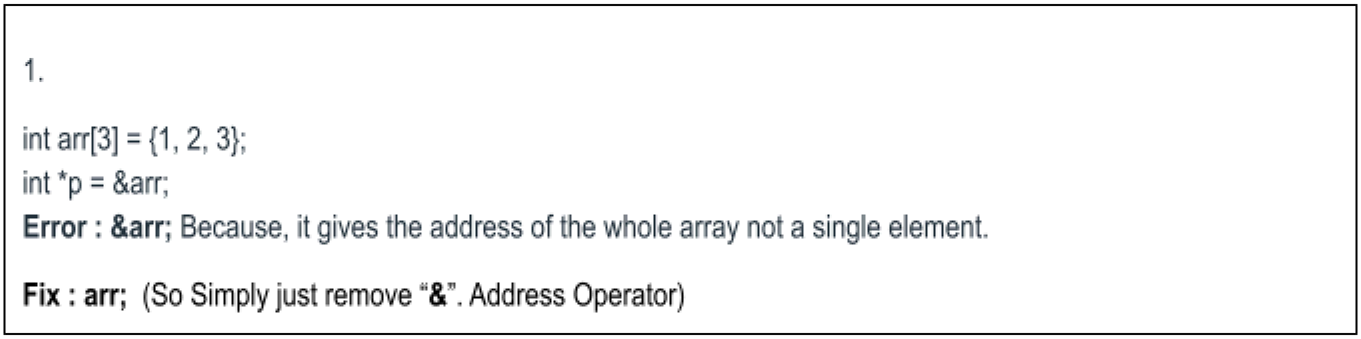


5.

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

OUTPUT : 25

Error Spotting :

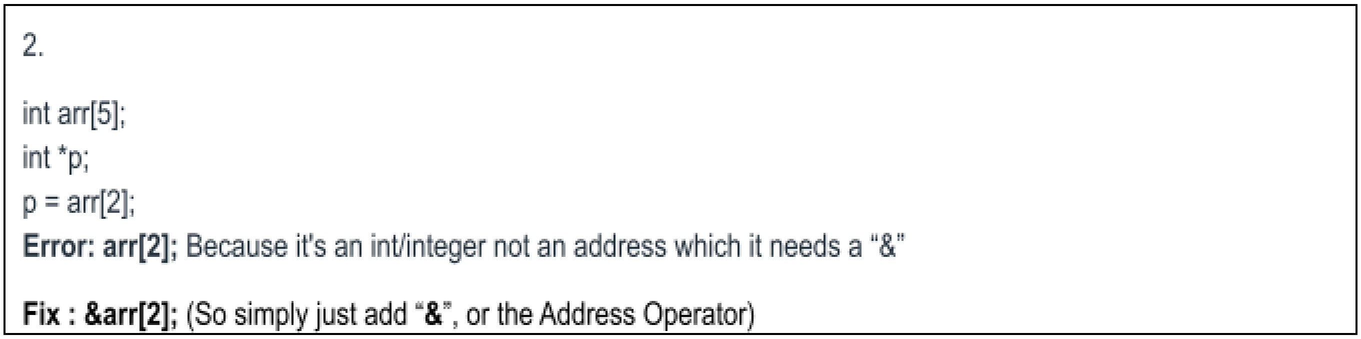


1.

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

Error : &arr; Because, it gives the address of the whole array not a single element.

Fix : arr; (So Simply just remove "**&**". Address Operator)

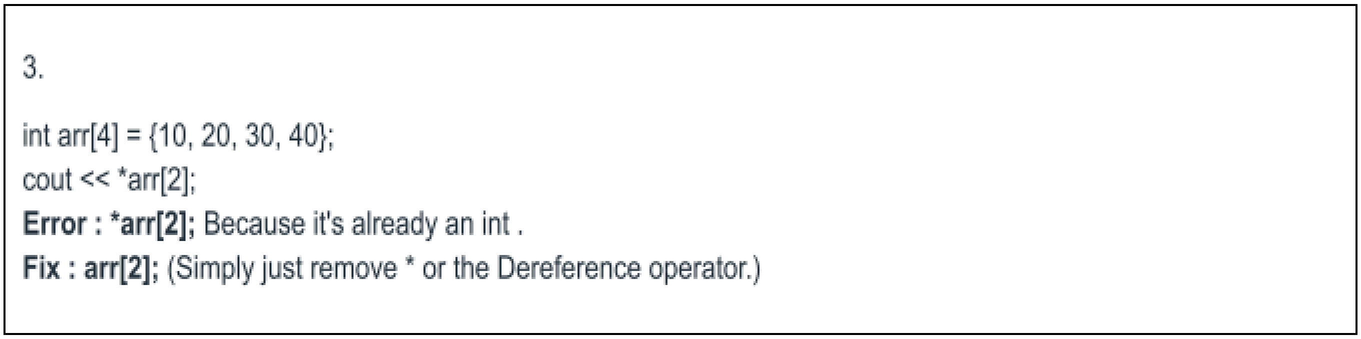


2.

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

Error: arr[2]; Because it's an int/integer not an address which it needs a "&"

Fix : &arr[2]; (So simply just add "**&**", or the Address Operator)



3.

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

Error : *arr[2]; Because it's already an int .
Fix : arr[2]; (Simply just remove * or the Dereference operator.)

8. Conclusion

In this assignment, it gives me another knowledge of C++ pointers that hold memory addresses of variables and additionally I gained more information on its uses, what it does, and when and why I should be implementing the Address Operators or the Deference Operators. Because, its uses showed me how it can manipulate memory directly and how it allows me to retrieve its value that is stored in a specific location. I also understood why pointers are important, such as for dynamic memory allocation and efficient array or function handling. Through the supplementary codes, I saw how pointers can point to variables and array elements, and how pointer arithmetic works when moving through arrays. Therefore that challenge allowed me to determine its possible outputs by knowing how those 2 operators function or work additionally, the recent lessons also contributed much to me while performing these activities because it allows me to understand what the other symbols do on coding. Lastly, in the error spotting part, I realized the importance of using the correct form when assigning addresses and dereferencing, since small mistakes can lead to invalid code or giving the wrong output. Overall, this strengthened my understanding of pointers, their uses, and their role in making C++ more powerful and flexible as this can additionally contribute as my reviewer to the upcoming quiz because I gained more information and the way of implementing pointers.