Lesson:

Pointers - 3

PW SKILLS

Pre-Requisites

- Basic understanding of pointers

List of Concepts Involved

- Types of pointers

Topic: Types of pointers

There are majorly 4 types of pointers-

1. Null pointer
2. Wild pointer
3. Dangling pointer
4. Void pointer

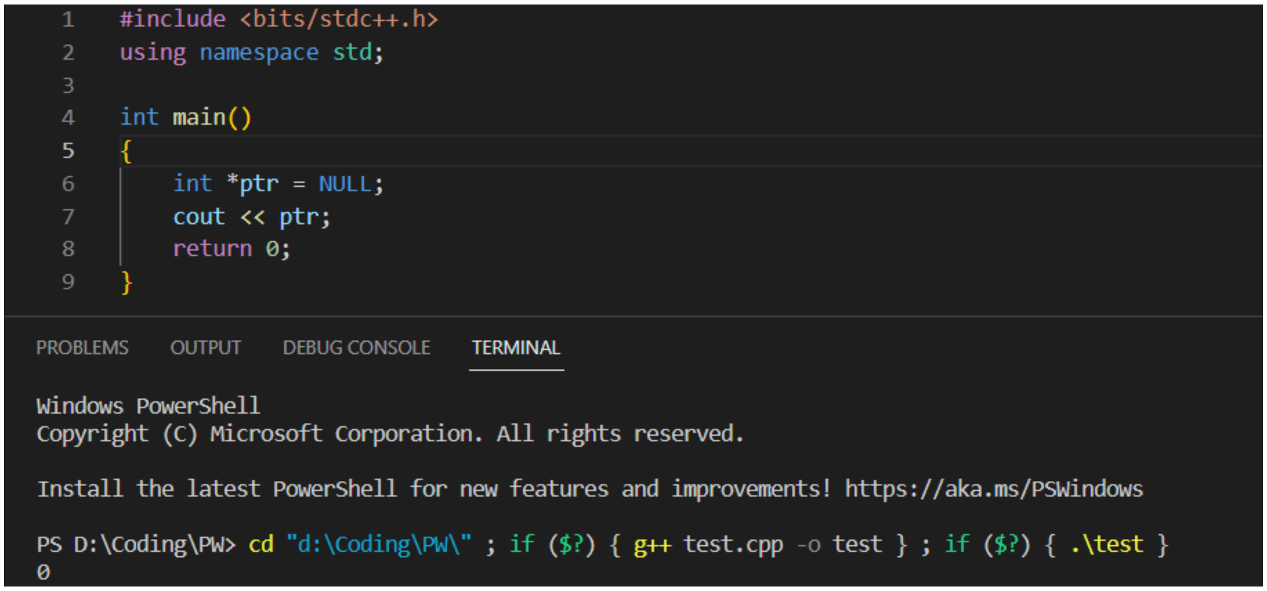
1. Null pointer

Just like other built-in types, pointers defined within the scope of a function or any other block of code have undefined value if not initialized. This could cause the program to read from or write to unexpected memory locations, leading to denial of service.

Thus it is often considered a good practice to initialize a pointer with 0 or NULL when the pointer doesn't point to a valid memory location. This is done because 0 is a reserved memory address in many operating systems, therefore a program is not allowed to access it. However, by convention, if a pointer is pointing to 0, it is assumed that it is pointing to nothing.

When we try to print the null pointer, we get the output as 0.

Code link - https://pastebin.com/J5gXn8SK



```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int *ptr = NULL;
    cout << ptr;
    return 0;
}
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\Coding\PW> cd "d:\Coding\PW\" ; if ($?) { g++ test.cpp -o test } ; if ($?) { .\test }
0
```

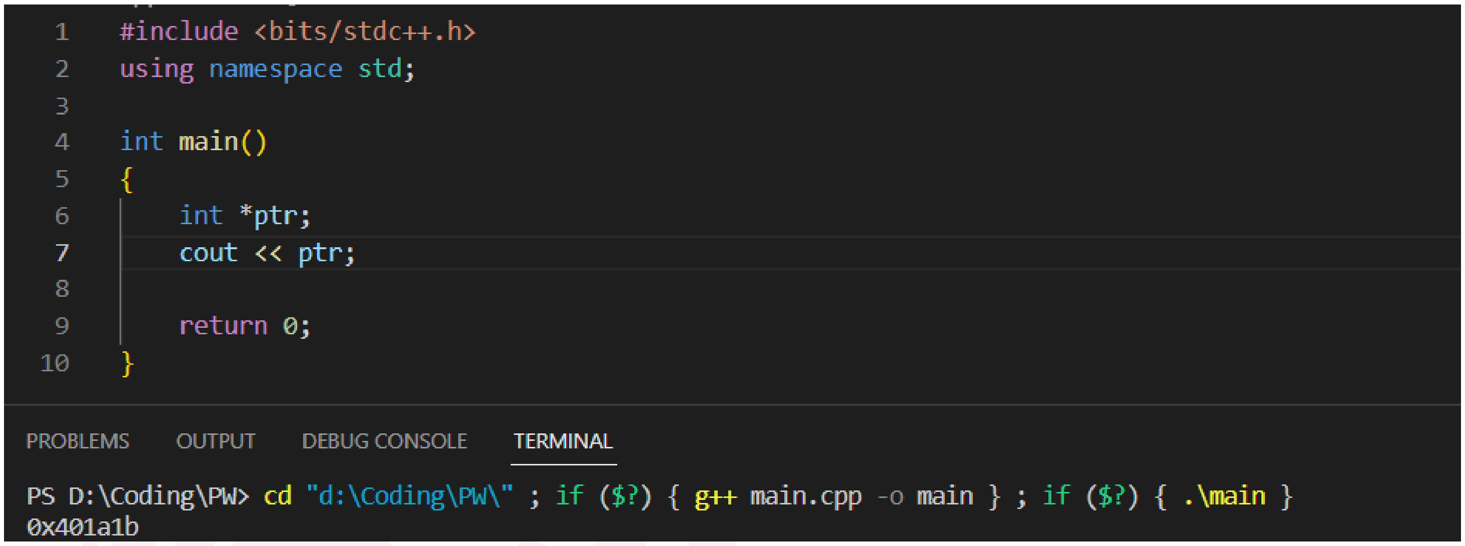
Note: When we try to deference a null pointer, we can often (not always) get a runtime error or the program may immediately crash.

2. Wild pointer

A wild pointer is a type of pointer where the user declares the pointer but doesn't initialise it with any value. Consequently they end up pointing to any random memory location. Using such pointers in a program causes the program to show undefined behavior or even crash. In some cases it also gives segmentation faults.

It is advised to use them cautiously.

Code link - https://pastebin.com/tWvFsQgN



```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int *ptr;
    cout << ptr;

    return 0;
}
```

```
PS D:\Coding\PW> cd "d:\Coding\PW\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
0x401a1b
```

3. Dangling pointer

It is the type of pointer that points to a memory location that is not valid.

There are 2 ways by which this can happen.

1. **Deallocation of memory**

Code link - https://pastebin.com/9pP4MHr1

In this example, malloc is used to dynamically allocate memory. It returns the address of the first byte of the memory location. We are storing that in the pointer ptr.In the second line, we are deallocating that memory. However the pointer ptr is still pointing to the de-allocated memory as we still haven't given it any other value. As a result we call this pointer a dangling pointer.

Thus it is important to re-initialize the pointer after freeing the memory.

2. Variable goes out of scope

Code link - https://pastebin.com/6QfawF4Z

In this example we are making ptr point to a variable that is defined with the scope of the curly braces. Now once we exit the scope, the variable gets deleted from the memory. However, we are not updating the value of the pointer, which means that it is still pointing to the same memory location. Consequently we call this pointer a dangling pointer.

4. Void pointer

These are special types of pointers that can point to an object of any type.

Code link - https://pastebin.com/5aatgF6Q

Upcoming topic

- Recursion