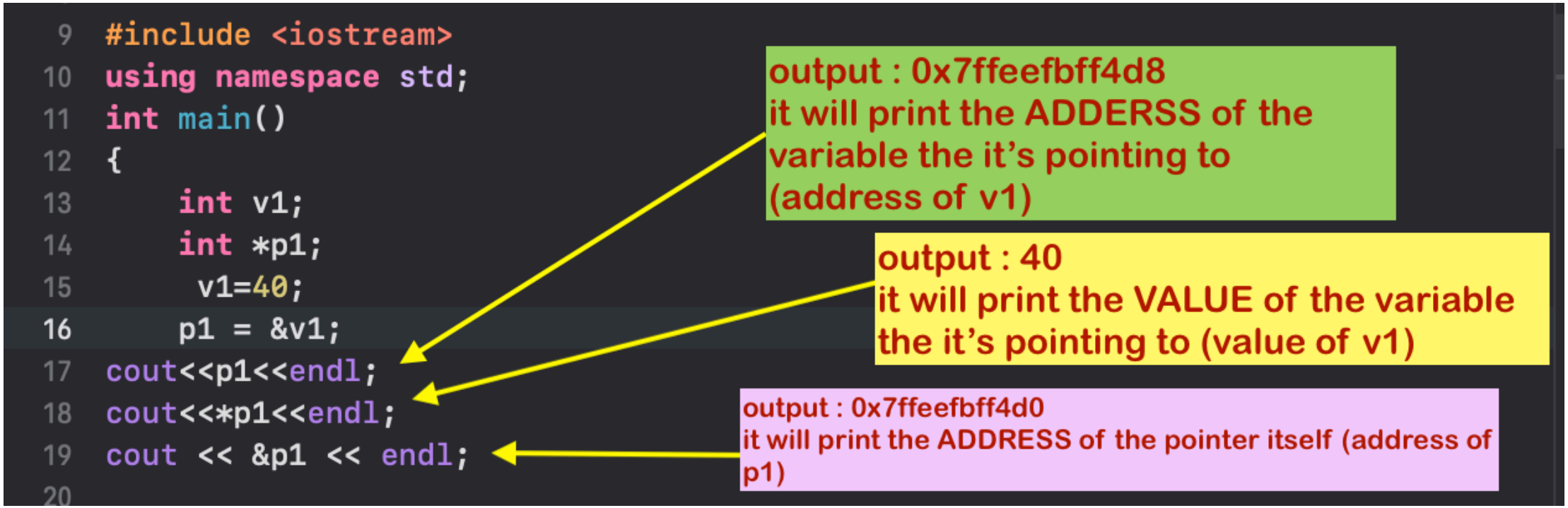
C++ Pointers

NoorHomaid


#include <iostream>
using namespace std;
int main()
{
int v1;
int *p1;
v1=40;
p1 = &v1;
cout<<p1<<endl;
cout<<*p1<<endl;
cout << &p1 << endl;
output : 0x7ffeefbff4d8
it will print the ADDERSS of the variable the it's pointing to (address of v1)
output : 40
it will print the VALUE of the variable the it's pointing to (value of v1)
output : 0x7ffeefbff4d0
it will print the ADDRESS of the pointer itself (address of p1)

Reference Vs. Pointers

Reference

-You need to initialize the reference during declaration

example

```
int &ref=3;
```

-You can not change the reference to reference another variable.

-To assign an address of a variable to a reference variable, no address-of operator & is needed

-No explicit dereferencing operator * should be used to return the value of the variable

Pointers

-To assign an address of a variable into a pointer, you need to use address-of operator & (e.g., pNumber = &number).

-You can change the pointer to point to another variable.

-To get the value pointed to by a pointer, you need to use the dereferencing operator *.

Dynamic variables.

Code examples

Simple example :

```
6  //  Copyright © 1441 nooralialhomaid. All rights reserved.
7  #include <iostream>
8  #include <cmath>
9  using namespace std;
10 int main(){
11     char *p; //p is a pointer to a char
12     p= new char; //p will point to a something of type char , yet it isn't
           pointing to any value
13     *p='E'; //now p is pointing the char E
14     cout << "P is pointnt to : " << *p << endl;
15 return 0;}
```

output :
p is pointing to E

Dangling pointer :

- Is a poiner that is UNDEFIEND

Meaning that if I deleted a dynamic variable , but I did not delete it's pointer in this case that pointer id undefiend (it's not pointing to anything)

This pointer

called **Dangling pointer**

```
6  //  Copyright © 1441 nooralialhomaid. All rights reserved.
7  #include <iostream>
8  using namespace std;
9  int main(){
10     int *pointer1;
11     pointer1= new int;
12     *pointer1=400;
13     delete pointer1; //i deleted the value 400 , so pointer1 is now pointing
           to nothing
14     //pointer1 is called dangling pointer
15     //to get rid of dangling pointer we simply delete it
16     pointer1=NULL;
17
18
19 return 0;}
20
```

How to delete a dynamic variable :

- By deleting the dynamic variable and it’s pointer

Delete (**pointer name**);

(**Pointer name**)=**NULL**;

According to the previous example :

Delete p ;

p= NULL ;

```
6  //  Copyright © 1441 nooralialhomaid. All rights reserved.
7  #include <iostream>
8  #include <cmath>
9  using namespace std;
10 int main(){
11     char *p;
12     p= new char;
13     *p='E';
14     cout << "P is pointnt to : " << *p << endl;
15     delete p;
16     p=NULL;
17
18 return 0;}
```

Defining Pointer Types

- To avoid mistakes using pointers, define a pointer type name
  - Example: typedef int* IntPtr;

    Defines a new type, IntPtr, for pointer variables containing pointers to int variables
  - IntPtr p;

  is equivalent to

  int *p;

Codes examples :

```cpp
//  Copyright © 1441 nooralialhomaid. All rights reserved.
#include <iostream>
using namespace std;
void print()
{
    int arr[3] = {22, 30, 100};
    int *p;
    p = arr;

    for (int i = 0; i < 3; i++)
    {
            cout << "Value at the pointer p = " << *p << "\n";
            cout << "The address of the value at p = " << p<< "\n";
            p++;
    }
}
  int main()
{
    print();
}
```

output:

```
Value at the pointer p = 22
The address of the value at p = 0x7ffeefbff4bc
Value at the pointer p = 30
The address of the value at p = 0x7ffeefbff4c0
Value at the pointer p = 100
The address of the value at p = 0x7ffeefbff4c4
```

```cpp
//  Copyright © 1441 nooralialhomaid. All rights reserved.
#include <iostream>
using namespace std;
void fun(int *p)
{
  int a = 10;
  p = &a;
}
int main()
{
  int x = 20;
  int *p = &x;
  fun(p);
cout<< *p<<endl;
  return 0;
}
```

output:
10

```cpp
//  Copyright © 1441 nooralialhomaid. All rights reserved.
#include <iostream>
# include <stdio.h>
using namespace std;
void fun(int *p1)
{
    *p1 = 10;
}
int main()
{
  int x = 20;
  fun(&x);
cout<<x<<endl;
  return 0;
}
```

output:
10