

# Fundamentals of Programming: Pointers in C++

Abdul Haseeb

# Agenda

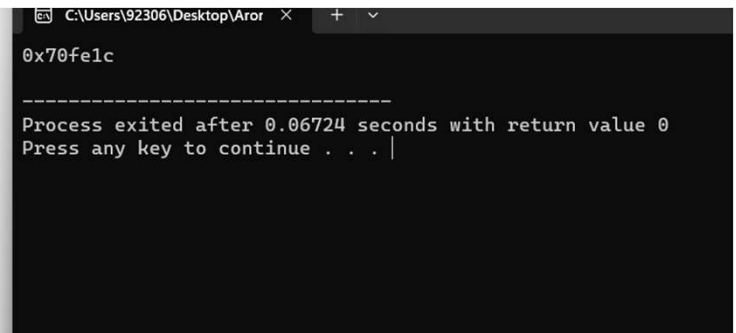
- Address Operator
- Definition of a pointer
- Syntax of a pointer
- Creating pointer variables and accessing those
- Use of Dereference Operator
- Pointer to Pointer
- Pointer Arithmetic (Increment/Decrement)

# Address Operator (&)

- ▶ Address operator returns the memory address of a variable.

```
#include<iostream>
using namespace std;

int main(){
    int a=5;
    cout<<&a<<endl;
    return 0;
}
```



The screenshot shows a Windows command prompt window titled "C:\Users\92306\Desktop\Aror". The output of the program is displayed as follows:

```
0x70fe1c
-----
Process exited after 0.06724 seconds with return value 0
Press any key to continue . . . |
```

```
#include<iostream>
using namespace std;

int main(){
int a=5;
int b=&a;
cout<<b<<endl;
return 0;
}
```

Can we  
do this....

# Pointer

5



A variable which holds the memory address of other variables of same data type.



Pointers are symbolic representation of memory addresses.

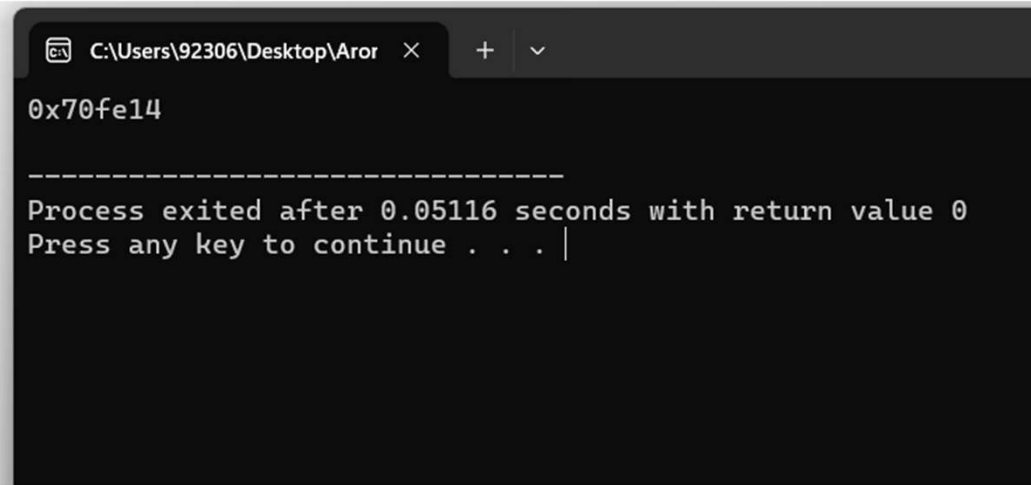
# Syntax

6

Syntax:

```
datatype *var_name;  
int *ptr;    // ptr can point to an address which holds int data
```

```
int main(){  
int a=5;  
int* b=&a;  
cout<<b<<endl;  
return 0;  
}
```

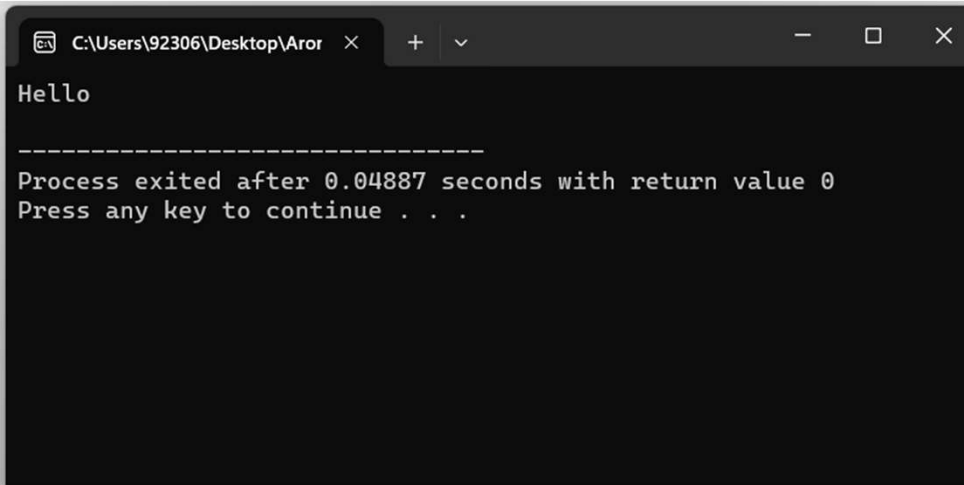


```
C:\Users\92306\Desktop\Aror  X + v  
0x70fe14  
-----  
Process exited after 0.05116 seconds with return value 0  
Press any key to continue . . . |
```

Pointer variable returns the address where variable a is stored in memory

```
#include<iostream>
using namespace std;

int main(){
string a="Hello";
string* b=&a;
cout<<*b<<endl;
return 0;
}
```

A screenshot of a Windows console window titled 'C:\Users\92306\Desktop\Aror'. The window shows the output of a C++ program. It first displays 'Hello' on a new line. After a series of dashes, it shows 'Process exited after 0.04887 seconds with return value 0' and 'Press any key to continue . . .'.

```
C:\Users\92306\Desktop\Aror  x  +  v  -  □  x
Hello

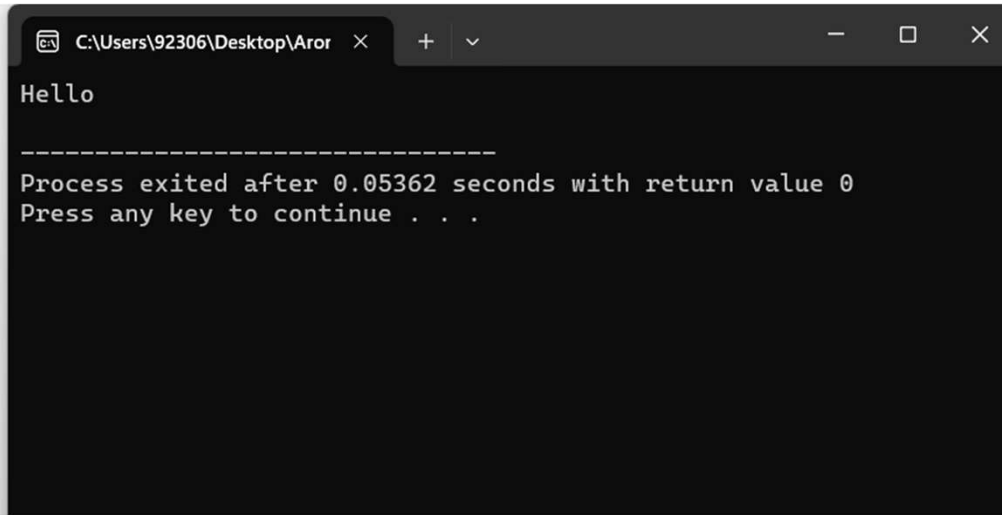
-----
Process exited after 0.04887 seconds with return value 0
Press any key to continue . . .
```

Another program, to illustrate the usage of dereference operator \*



```
#include<iostream>
using namespace std;

int main(){
string a="Hello";
string* b=&a;
cout<<*&a<<endl;
return 0;
}
```

A screenshot of a Windows console window titled "C:\Users\92306\Desktop\Aror". The window shows the output of a C++ program. It first displays "Hello" on a new line. After a series of dashes, it shows "Process exited after 0.05362 seconds with return value 0" and "Press any key to continue . . .".

```
C:\Users\92306\Desktop\Aror  x  +  v  -  □  x
Hello

-----
Process exited after 0.05362 seconds with return value 0
Press any key to continue . . .
```

Another program, to illustrate the usage of dereference operator \*

# Dereference operator (\*)

IT GIVES THE VALUE STORED AT ANY MEMORY ADDRESS.

## Variety of uses...

- ▶ Call by reference
- ▶ Iterating over arrays
- ▶ Manipulating dynamic arrays

# Why to associate data type with a pointer, Why not directly store address

- ▶ Data type tells us how many bytes the data is stored in.
- ▶ When we increment a pointer, we increase the pointer by the size of data type it points to.

# Pointer to Pointer

```
int main(){  
    string a="Hello";  
    string* b=&a;  
    string**c =&b;  
  
    cout<<&a<<endl;  
    cout<<b<<endl;  
    cout<<*b<<endl;  
    cout<<&b<<endl;  
  
    cout<<c<<endl;  
    cout<<*c<<endl;  
    cout<<**c;
```

```
C:\Users\92306\Desktop\Aror  
0x71fdf0  
0x71fdf0  
Hello  
0x71fde8  
0x71fde8  
0x71fdf0  
Hello  
-----  
Process exited after 0.  
Press any key to contin
```

# Pointer Arithmetic (Increment/Decrement)

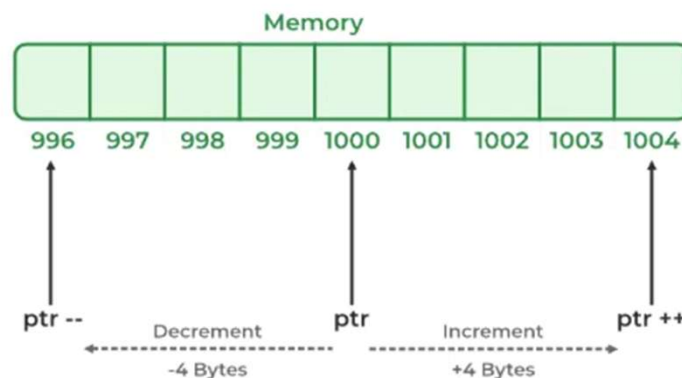
14

- ▶ Applying increment/decrement operator on pointer variable will change the memory address

# Pointer Arithmetic (Addition)

15

## Pointer Increment & Decrement



# Increment in Pointer

16

```
#include<iostream>
using namespace std;

int main(){

int a=5;
int* ptr=&a;
cout<<ptr<<endl;
cout<<++ptr<<endl;
return 0;
}
```

```
C:\Users\92306\Desktop\Aror  ×  +  v
0x70fe14
0x70fe18

-----
Process exited after 0.08117 seconds with return value 0
Press any key to continue . . .
```



# What about this:

17

```
#include<iostream>
using namespace std;

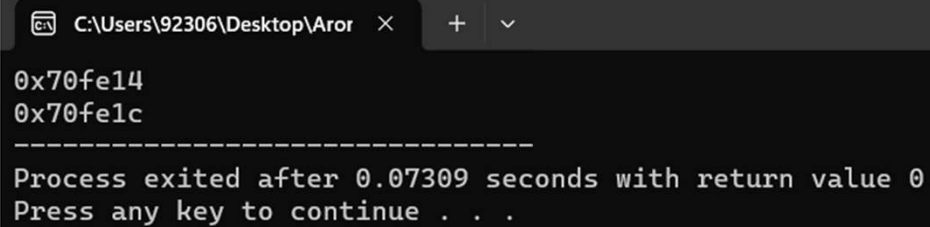
int main(){
int a=5;
int* ptr=&a;
cout<<ptr<<endl;
cout<<++ptr<<endl;
cout<<++ptr;
return 0;
}
```

# Further on Increment:

18

```
#include<iostream>
using namespace std;

int main(){
int a=5;
int* ptr=&a;
cout<<ptr<<endl;
cout<<ptr+2;
}
```



A screenshot of a terminal window showing the execution of a C++ program. The window title is "C:\Users\92306\Desktop\Aror". The output shows two memory addresses: "0x70fe14" and "0x70fe1c", separated by a line of dashes. Below the dashes, it says "Process exited after 0.07309 seconds with return value 0" and "Press any key to continue . . .".

```
C:\Users\92306\Desktop\Aror  ×  +  ▾
0x70fe14
0x70fe1c
-----
Process exited after 0.07309 seconds with return value 0
Press any key to continue . . .
```

# Pointer Decrement

19

```
#include<iostream>
using namespace std;

int main(){
int a=5;
int* ptr=&a;
cout<<ptr<<endl;
cout<<--ptr;
}
```

```
C:\Users\92306\Desktop\Aror  X + v
0x70fe14
0x70fe10
-----
Process exited after 0.06461 seconds with return value 0
Press any key to continue . . .
```

# Null Pointer

20

- ▶ A Pointer which points no where
- ▶ If we don't have address to be assigned to a pointer, then we can simply use NULL.
- ▶ Two ways to assign a pointer as null:
  - ▶ `int *ptr1=0;`
  - ▶ `int *ptr2=NULL;`

# Null vs Uninitialized Pointer

21

- ▶ An uninitialized pointer stores an undefined value.
- ▶ A null pointer stores a defined value, but one that is defined by the environment to not be a valid address for any member or object.