Intro to C++ | 1.2

God Bless Pointers

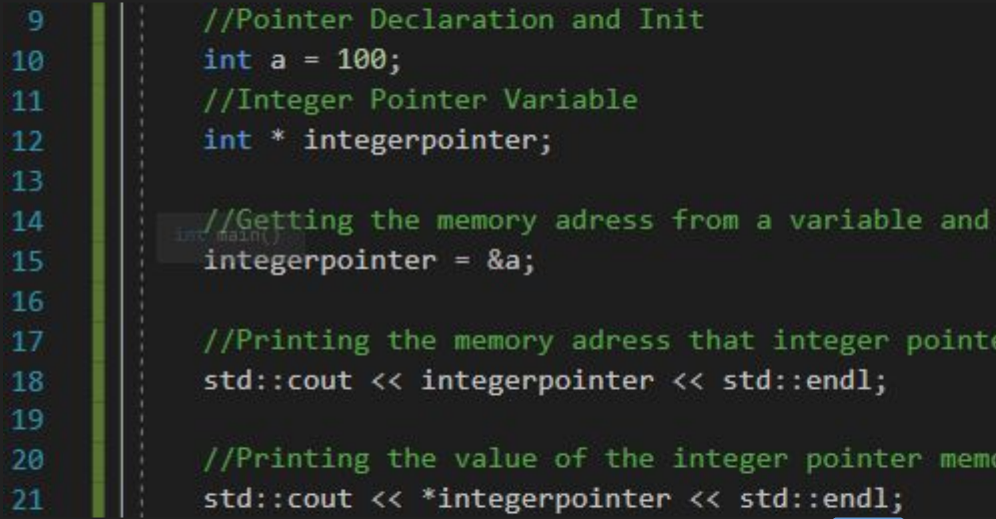
Akis Zachariadis - Games Dept. Coordinator

Pointers

- It is a specialised type of variable that holds the memory address of a regular variable.

Declaration | *,&

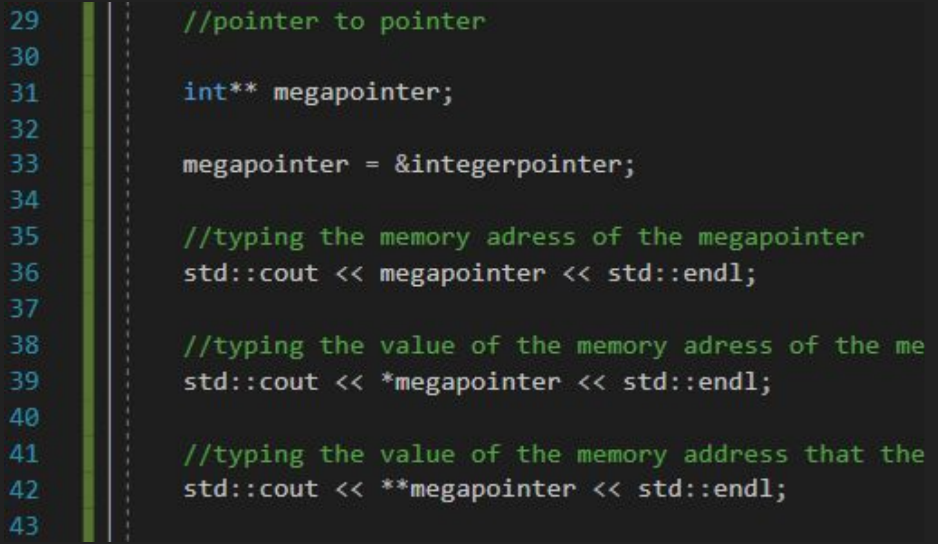
- Line 12 | Declaration of an pointer type of integer
- Line 15 | Init of the pointer. Getting the memory address of the a variable
- Line 17 | Print the memory address that the pointer holds
- Line 20 | Printing the value of the memory adress



```
 9      //Pointer Declaration and Init
10      int a = 100;
11      //Integer Pointer Variable
12      int * integerpointer;
13
14      //Getting the memory adress from a variable and
15      integerpointer = &a;
16
17      //Printing the memory adress that integer pointe
18      std::cout << integerpointer << std::endl;
19
20      //Printing the value of the integer pointer mem
21      std::cout << *integerpointer << std::endl;
```

Pointers to pointers **

- Line 31 | Declaration of an double point type of integer
- Line 33 | Getting the memory address from the integer pointer
- Line 38 | Print the value of the megapointer which is the memory address of the integer pointer
- Line 41 | Printing the value of the the value of the integer pointer. This will print 100

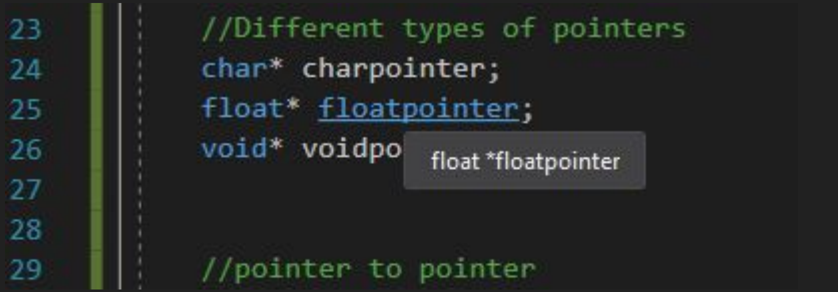


```
29      //pointer to pointer
30
31      int** megapointer;
32
33      megapointer = &integerpointer;
34
35      //typing the memory adress of the megapointer
36      std::cout << megapointer << std::endl;
37
38      //typing the value of the memory adress of the me
39      std::cout << *megapointer << std::endl;
40
41      //typing the value of the memory address that the
42      std::cout << **megapointer << std::endl;
43
```

Type of pointerts

- Char
- Float
- Double
- Integer
- Void

- No string type

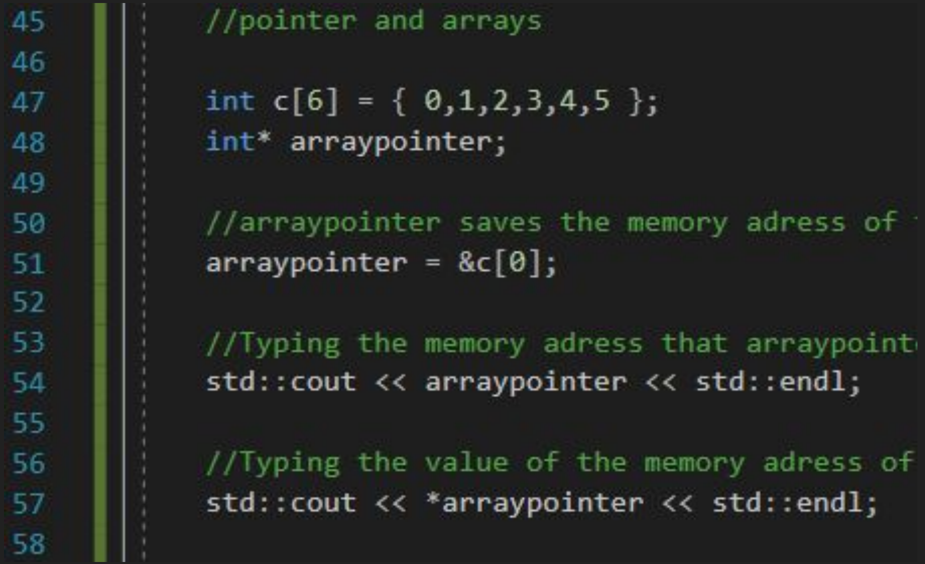


```
//Different types of pointers
char* charpointer;
float* floatpointer;
void* voidpo

//pointer to pointer
```

Pointers and Arrays

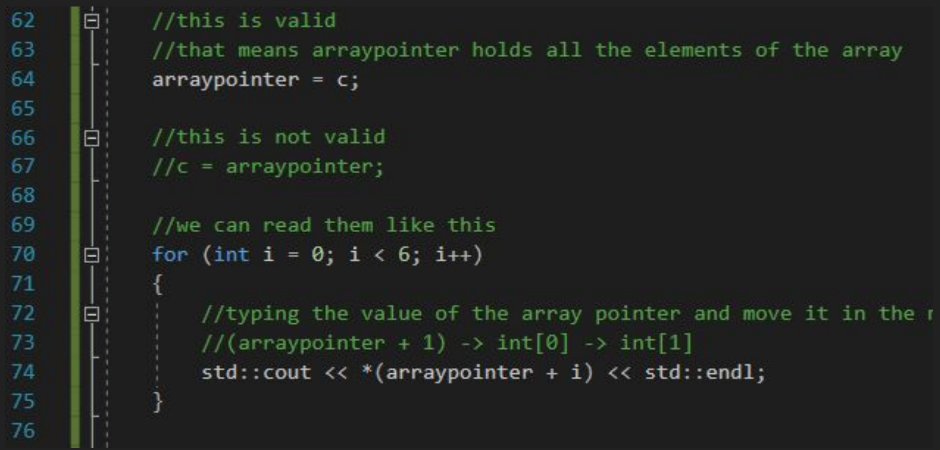
- Line 47 | Create an integer array
- Line 48 | Create an integer pointer
- Line 50 | Getting the memory address of the first element of the array
- Line 54-57 | Print the memory address and value of the pointer



```
45      //pointer and arrays
46
47      int c[6] = { 0,1,2,3,4,5 };
48      int* arraypointer;
49
50      //arraypointer saves the memory adress of
51      arraypointer = &c[0];
52
53      //Typing the memory adress that arraypoint
54      std::cout << arraypointer << std::endl;
55
56      //Typing the value of the memory adress of
57      std::cout << *arraypointer << std::endl;
58
```

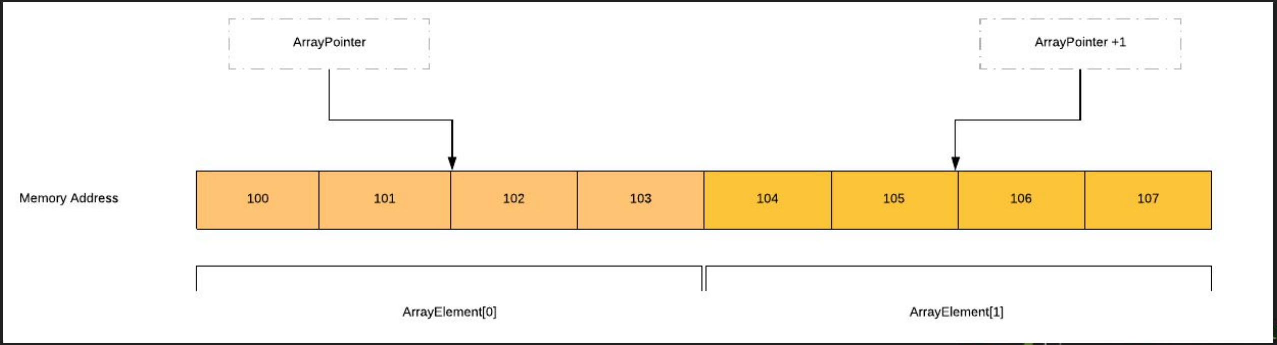
Arrays are pointers

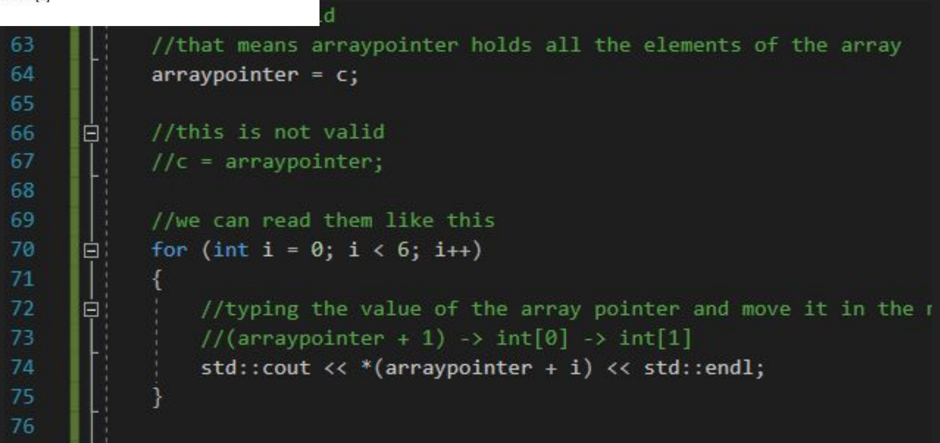
- Line 64 | We can directly assign an array to a pointer. That means pointer holds all the memory addresses of the array.
- Line 70 | That means we can read the array through the pointer using a for loop and hoping through memory chunks



```
62  //this is valid
63  //that means arraypointer holds all the elements of the array
64  arraypointer = c;
65
66  //this is not valid
67  //c = arraypointer;
68
69  //we can read them like this
70  for (int i = 0; i < 6; i++)
71  {
72      //typing the value of the array pointer and move it in the r
73      //(arraypointer + 1) -> int[0] -> int[1]
74      std::cout << *(arraypointer + i) << std::endl;
75  }
76
```

Arrays are pointers





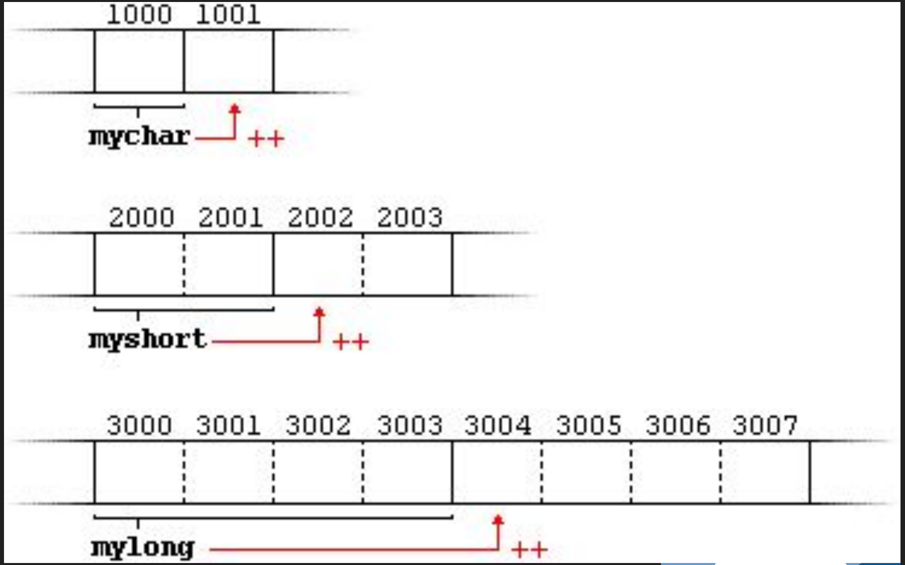
```
//that means arraypointer holds all the elements of the array
arraypointer = c;

//this is not valid
//c = arraypointer;

//we can read them like this
for (int i = 0; i < 6; i++)
{
    //typing the value of the array pointer and move it in the
    //(arraypointer + 1) -> int[0] -> int[1]
    std::cout << *(arraypointer + i) << std::endl;
}
```

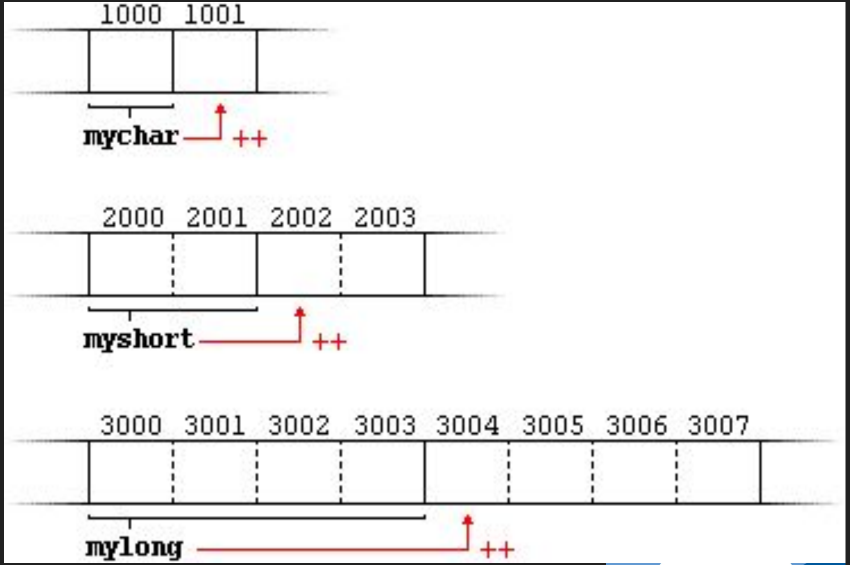
Arrays are pointers | Memory Chunks

- Depending the type of pointer (int) it changes how the pointer represents values to the memory.
- myLong variable is a type of integer and it is 32bit long. Most of the times memory is splitted to 8bit of chunks. That means for an integer variable we need 4 chunks.
- When we say integer* + 1 means move 4 chunks to the right in the memory



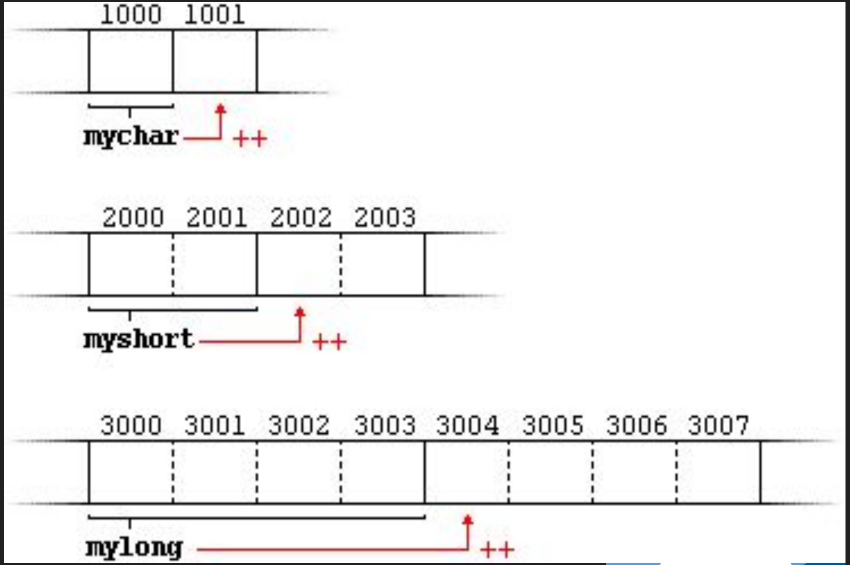
Arrays are pointers | Memory Chunks

- If the pointer is a type of char then it only needs 8-bit for a single char. That means 1 chunk of memory.
- We must be careful with the type of pointer and how the specific type holds its memory.
- Array holds its memory address in a sequence that why its size is not dynamic
- Vector holds its memory address in a sequence way until we grow the array.

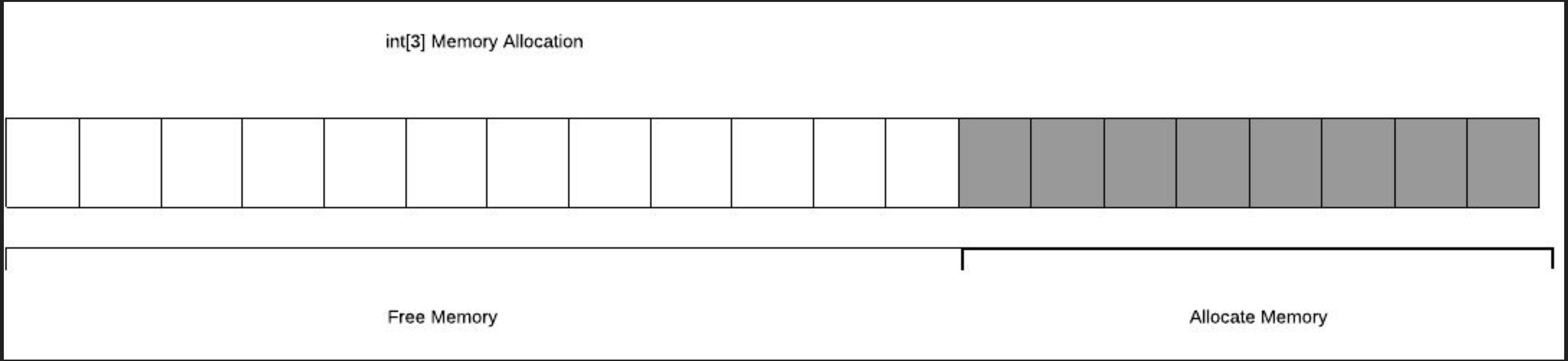


Arrays are pointers | Memory Chunks

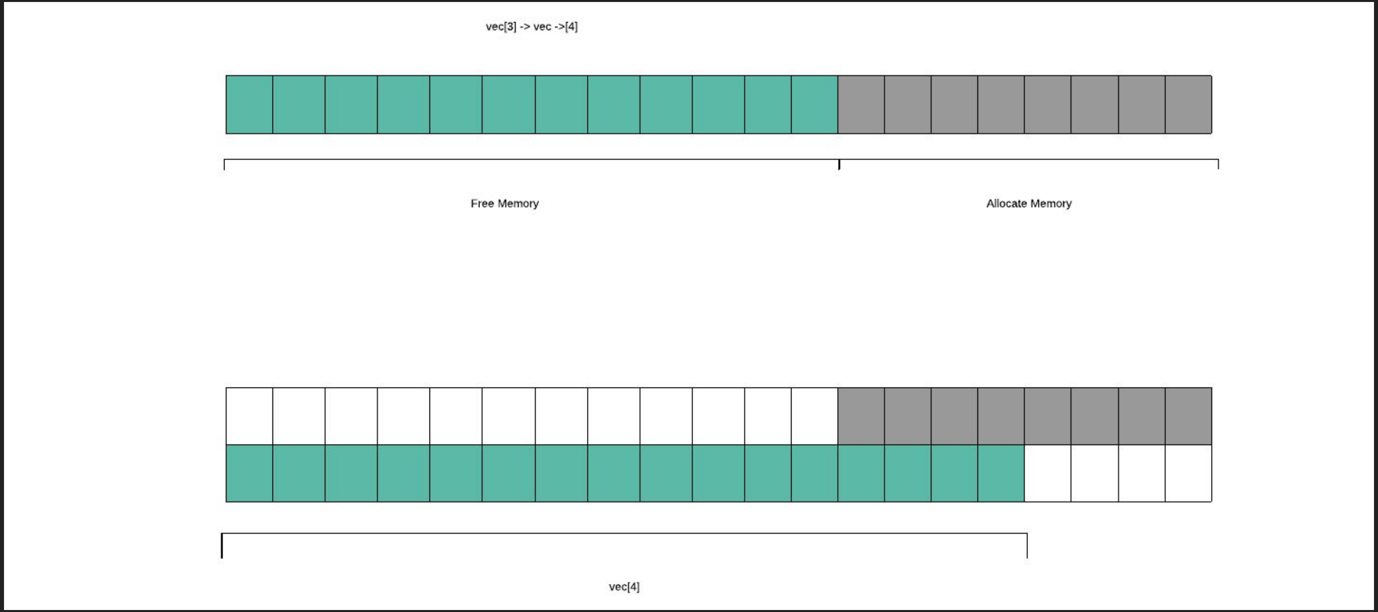
- If the vector implementation needs more memory it will move all the elements to a different part of the memory.
- Our pointer will still points to the previous memory address and that means to nowhere or something completely different.



Arrays are pointers | Memory Chunks

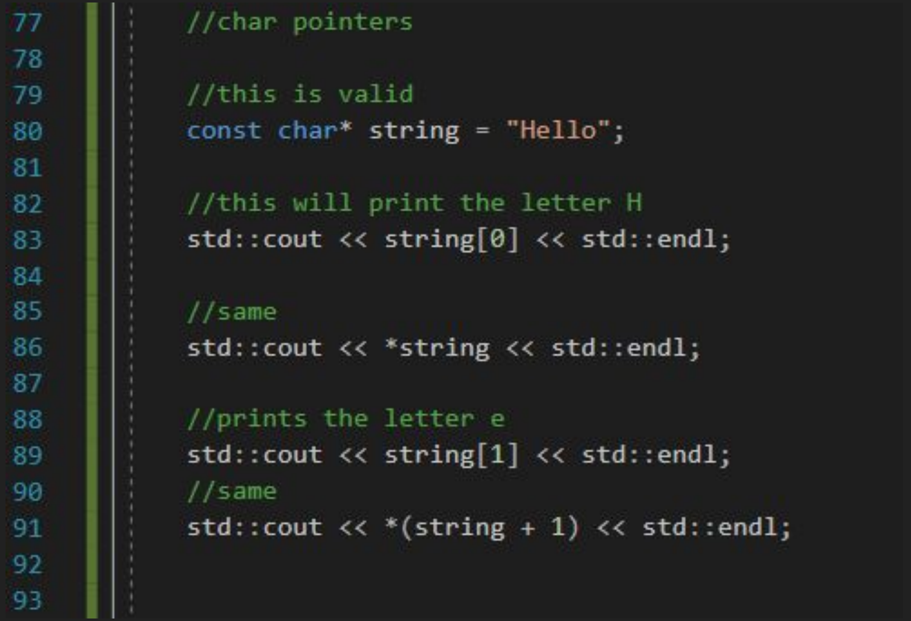


Arrays are pointers | Memory Chunks



Char Pointers

- Line 80 | Create a char pointer and assign a string.
- This will lead to a char pointer array with values the string value
- If we try to print the value of the pointer it will print the first value of string. “H”
- If we hop in the memory to the right then we will print the rest of the elements
- Most string variables are actually char[] pointers.



```
77  //char pointers
78
79  //this is valid
80  const char* string = "Hello";
81
82  //this will print the letter H
83  std::cout << string[0] << std::endl;
84
85  //same
86  std::cout << *string << std::endl;
87
88  //prints the letter e
89  std::cout << string[1] << std::endl;
90  //same
91  std::cout << *(string + 1) << std::endl;
92
93
```

Challenge #1

Να δημιουργηθεί πρόγραμμα που να καλωσορίζει τον χρήστη, και θα τον καθοδηγεί για να επιλέξει το κατάλληλο χρώμα LightSaber ανάλογα με τα στοιχεία του χαρακτήρα του.

Αναλυτικότερα το πρόγραμμα θα πρέπει να :

- Να εμφανίζει εισαγωγικό μήνυμα για τη λειτουργία του προγράμματος.
- Να ζητάει όνομα από το χρήστη.
- Να ρωτάει τον χρήστη ποιός είναι ο χαρακτήρας του με βάση τα παρακάτω
  - Prefer negotiation and meditation, strong force abilities (Green)
  - Always angry and wants to destroy EVERYTHING (Red)
  - Jedi Guardians fighting for the light side and are skillful swordsmen (Blue)
  - Using Light and Dark side force, doing everything in order to keep the balance (Purple) (Chaotic Good).
- Να εμφανίζει τι χρώμα θα έχει το Lightsaber του χρήστη καθώς και το όνομά του.

Για την υλοποίηση να χρησιμοποιηθούν μόνο pointers.

Challenge #2

Για την συγκεκριμένη άσκηση θα δημιουργήσετε έναν τηλεφωνικό κατάλογο ως πρόγραμμα κονσόλας. Το πρόγραμμα αρχικά θα καλωσορίζει τον χρήστη και κατόπιν θα των ρωτάει ποιος είναι ο (σταθερός) αριθμός των καταχωρήσεων.

Κάθε καταχώρηση περιέχει ένα όνομα και ένα τηλέφωνο.

Αρχικά ο κατάλογος είναι κενός. Στην συνέχεια, το πρόγραμμα θα εμφανίζει ένα μενού με τις βασικές επιλογές: εμφάνιση του καταλόγου, εισαγωγή/μεταβολή καταχώρησης, διαγραφή καταχώρησης, αναζήτηση με βάση το όνομα, έξοδος.

Για την υλοποίηση θα χρησιμοποιήσετε δύο arrays, ένα string[] και ένα int[], για τα ονόματα και τα τηλέφωνα, αντίστοιχα.

Να χρησιμοποιηθούν μόνο οι pointers για να γίνει το manipulation arrays και μεταβλητών.