Introduction to C & C++

Assignments 02: Pointer

JJCAO

Pointer manipulation

Assume that the following variable declaration has already been made:

```
char *oddOrEven = “Never odd or even”;
```

Write a single statement to accomplish each of the following tasks (next page) (assuming that the previous ones have already been run). Save all statements in a win32 console application.

1.Create a pointer to a char value named nthCharPtr pointing to the 6th character of oddOrEven (remember that the first item has index 0). Use the indexing operator.

2. Using pointer arithmetic, update nthCharPtr to point to the 4th characterin oddOrEven.

3. Print the value currently pointed to by nthCharPtr.

4. Create a new pointer to a pointer (a char **) named pointerPtr that points to nthCharPtr.

5. Print the value pointed by pointerPtr. (tip: use printf() or cout << static_cast<void*>)

6. Print the value of pointerPtr

7. Using pointerPtr, print the value pointed to by nthCharPtr.

8. Update nthCharPtr to point to the next character in oddOrEven (i.e. one character past the location it currently points to).

9. Using pointer arithmetic,print out how faraway from the character currently pointed to by nthCharPtr is from the start of the string.

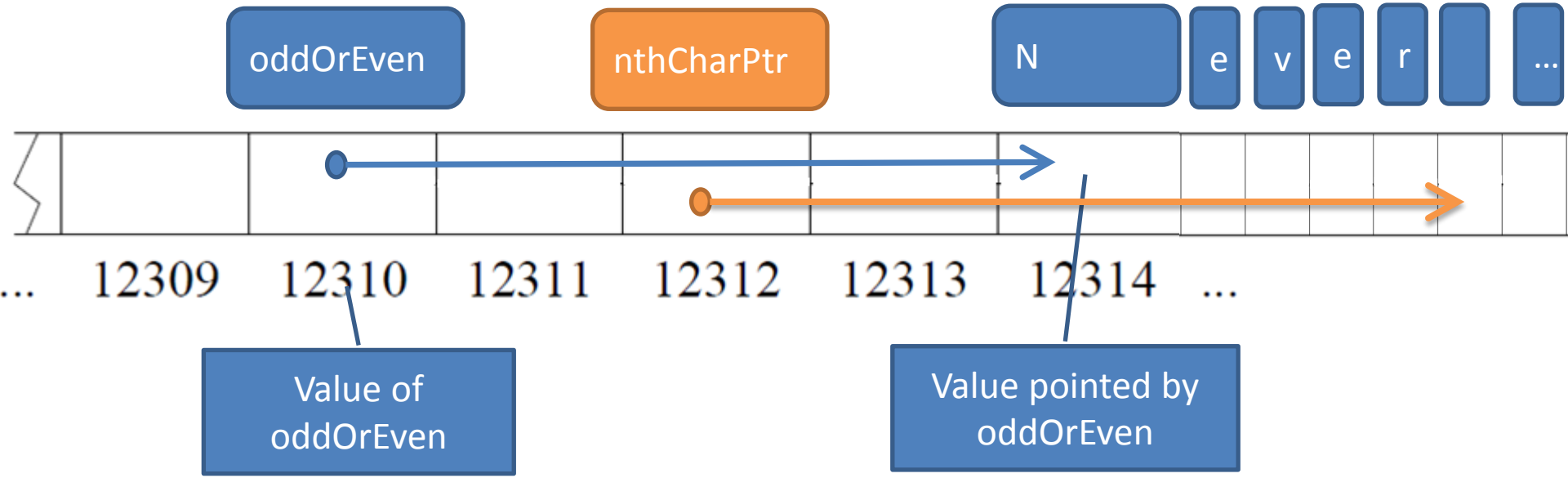
Notes

1. Print of char* (The C++ Standard Library: **13.3.3 Input/Output of Special Types)**
    1. char* cstring = "hello";
    2. cout << "string \"" << cstring << "\" is located at address: “ << static_cast<void*>(cstring) << endl;
2. If you do not understand, translate it and debug it!

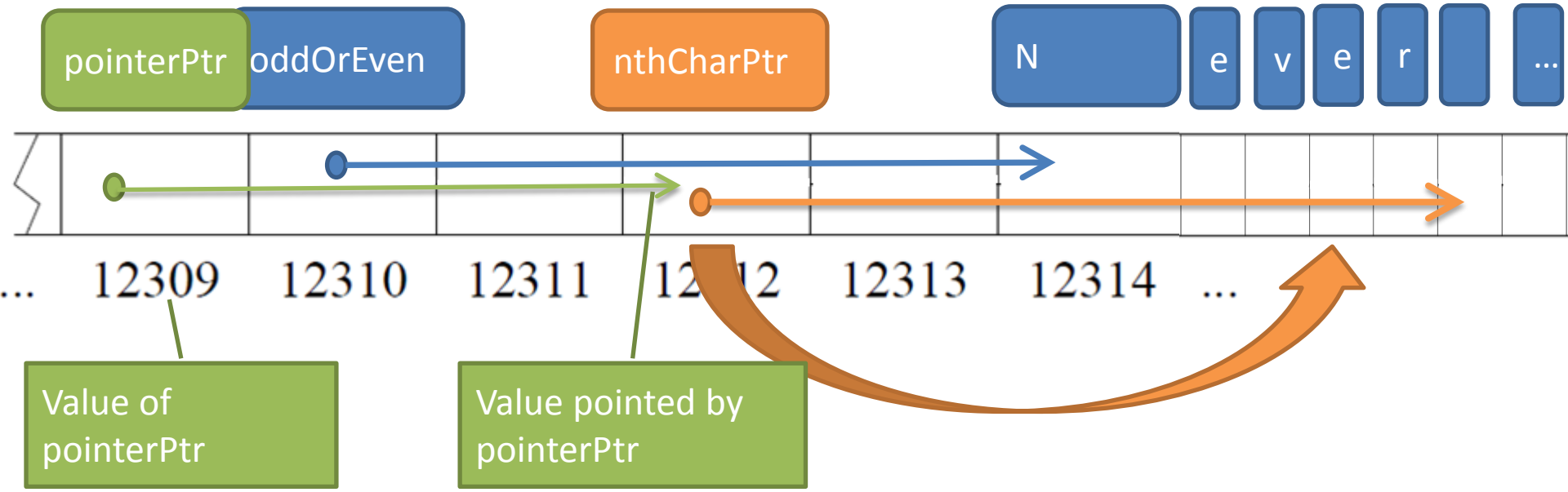
L-value & R-value

Literal Constants
R-value

```
char *oddOrEven = "Never odd or even";
```

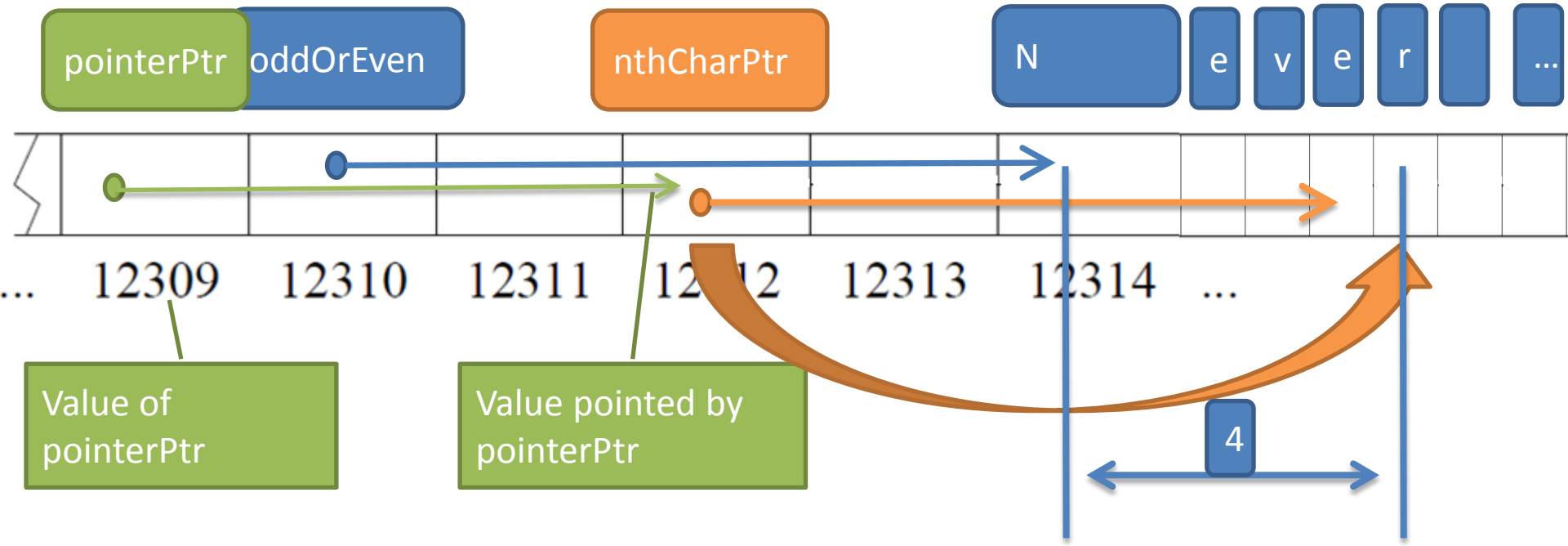



```
char *nthCharPtr = &oddOrEven[5];
```




```
nthCharPtr = nthCharPtr -2;
cout << "value pointed to by nthCharPtr: " << *nthCharPtr << endl;
char **pointerPtr = &nthCharPtr;

//5. Print the value pointed by pointerPtr.
printf("value pointed by pointPtr: %p\n", *pointerPtr);
cout << "value pointed by pointerPtr: " << static_cast<void*>(*pointerPtr) << endl;
//cout << "value pointed by pointerPtr: " << *pointerPtr << endl; // wrong
// 6. Print the value of pointerPtr;
cout << "value of pointerPtr: " << pointerPtr << endl;
```




```
//7. Using pointerPtr, print the value pointed to by nthCharPtr.
cout << "value pointed to by nthCharPtr, using pointerPtr: " << **pointerPtr << endl;

//8. Update nthCharPtr to point to the next character in oddOrEven (i.e. one character past the location it currently points to).
++nthCharPtr;

//9. Using pointer arithmetic, print out how far away from the character currently pointed to by nthCharPtr is from the start of the string.
cout << "distance from nthCharPtr to start of oddOrEven: " << nthCharPtr - oddOrEven << endl;
```