Exercise 1



The output of the code is 2 21 3. When we have twist(t, s) the value of r didn’t get touch at all, so it stays the same. Also, the value of t doesn’t change since t is an int meaning it doesn’t change and only int& can change like the value of s because it can access the memory location and make the value of s to change once being by update by the function twist.

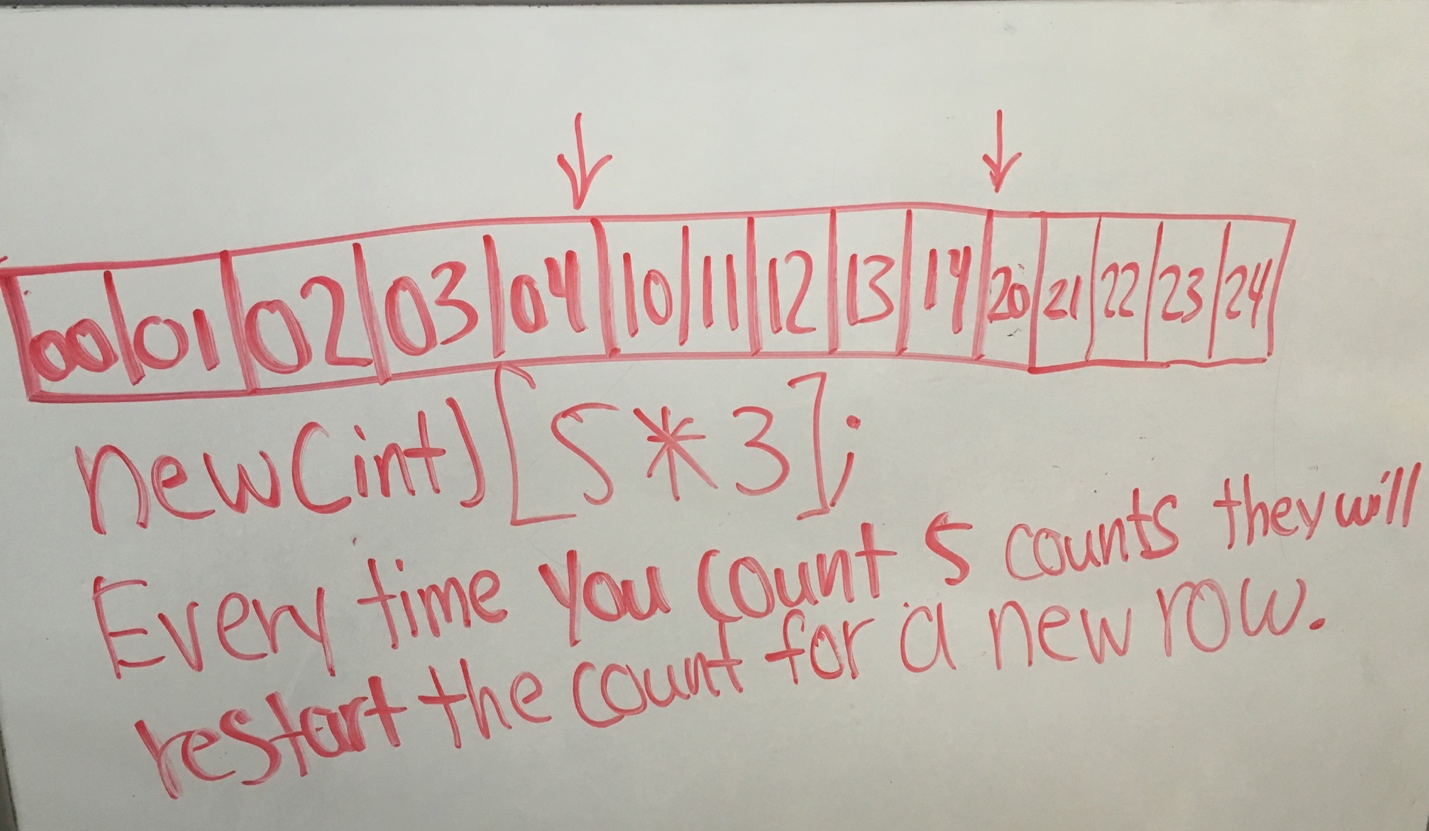


Exercise 2

1. If you have a static 2D array, the function doesn’t know size of the column. It does turn it into one dimension then the memory knows where to look for based on the information from the column of the memory then it would know where the rows would in in 1D array.



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 00 | 01 | 02 | 03 | 04 |
| 10 | 11 | 12 | 13 | 14 |
| 20 | 21 | 22 | 23 | 24 |





To find out the address of statement A[i][j] we can write the code for this problem, which is A[i \* cols + j] because when we put the number of columns for the user needs the system would go to that location and pick up the address of A[i][j].

Exercise 4

#include <iostream>

//we are missing using namespace std

int main()

{

int\* ptr;

int\* temp;

int x;

ptr = new int;

\*ptr = 4;

// we need temp to be allocated before it can be assigned by value

\*temp = \*ptr;

cout << ptr << temp; // need << “ ” after ptr and << endl after the temp so it makes the output is easier to see

x = 9;  
 \*temp = x;

cout << \*ptr << \*temp; // need << ‘ ‘ after ptr and << endl after the temp so it makes the output is easier to see

ptr = new int;

ptr = 5; //We need to change ptr to \*ptr because integer can’t convert into int\* assignment

cout << \*ptr << \* temp; // need << ‘ ‘ after ptr and << endl after the temp so it makes the output is easier to see // output is: …

return 0;

}

Correcting the code

#include <iostream>

//we are missing using namespace std

int main()

{

int\* ptr;

int\* temp;

int x;

ptr = new int;

\*ptr = 4;

temp = new int;

\*temp = \*ptr;

cout << ptr << ‘ ‘ << temp << endl;

x = 9;  
 \*temp = x;

cout << \*ptr << ‘ ‘ << \*temp << endl;

ptr = new int;

\*ptr = 5;

cout << \*ptr << ‘ ‘ << \*temp << endl; // output is: …

return 0;

}

Output:

0xd5fc20 0xd5fc40

4 9

5 9