

MODEL OF MEMORY

C++ ARRAYS

Problem Solving with Computers-I

<https://ucsb-cs16-sp17.github.io/>

C++

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

int main(){
    cout<<"Hola Facebook!n";
    return 0;
}
```



Reflecting on the midterm

- The question paper is on the course website: <https://ucsb-cs16-sp17.github.io/exam/e01/>
- Lab04 is now available – all about arrays!
- Hw08 is also all about arrays and tracing code!

Memory and C++ programs

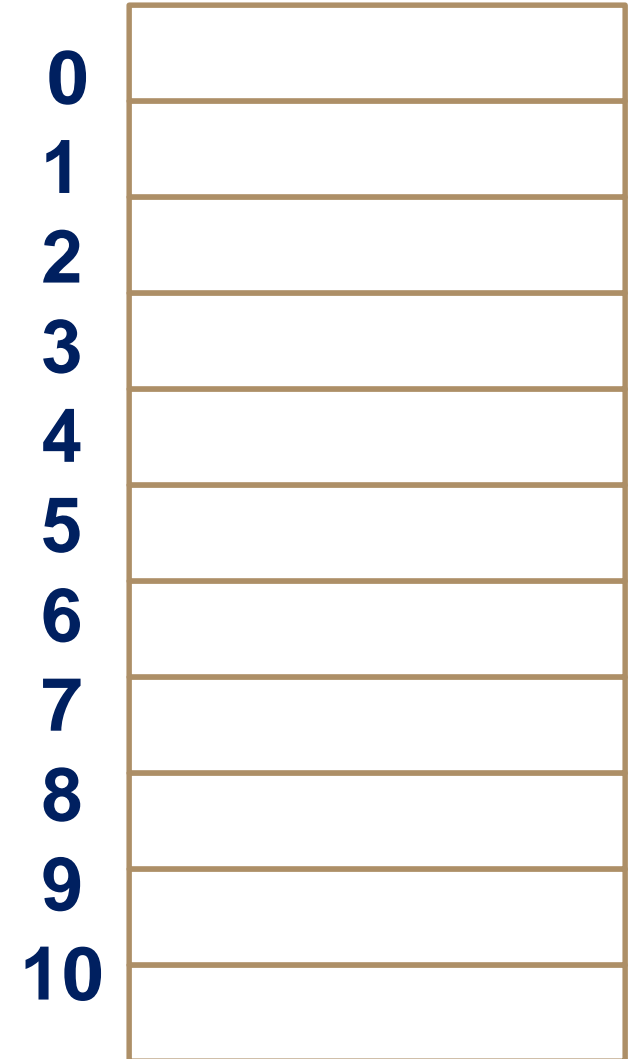
“The overwhelming majority of program bugs and computer crashes stem from problems of memory access... Such memory-related problems are also notoriously difficult to debug. Yet the role that memory plays in C and C++ programming is a subject often overlooked.... Most professional programmers learn about memory entirely through experience of the trouble it causes.”

.... Frantisek Franek
(Memory as a programming concept)

Model of memory

- Sequence of adjacent cells
- Each cell has 1-byte stored in it
- Each cell has an address (memory location)

```
char x = 1;  
int y = 4;  
char tmp = x;  
x = y;  
y = tmp;
```



Array motivation

- Write a program to record the midterm scores of 10 students in CS16, by asking the user to input each score. Then print out each of the recorded scores

C++ Arrays

A C++ array is a **list of elements** that share the same name, have the same data type and are located adjacent to each other in memory

scores

10
20
30
40
50
60
70
80

10	20	30	40	50	60	70	80
----	----	----	----	----	----	----	----

scores

What is the memory location of each element?

scores

10	20	30	40	50
----	----	----	----	----

```
int scores[5]={10, 20, 30, 40, 50};
```

If the starting location of the array is 0x200, what is memory location of element at index 2?

- A. 0x201
- B. 0x202
- C. 0x204
- D. 0x208

Declaring C++ arrays



```
int scores[5]; // declares a 5-element integer array  
                //declare a 5-element char array
```


Declaring and initializing, accessing elements



// Declare a 5-element integer array and fill it with values

```
int scores[5]={10, 20, 30, 40, 50};
```

Exercise: Reassign each value to 60



scores[0] scores[1] scores[2]

```
int scores[]={20,10,50}; // declare and initialize  
//Access each element and reassign its value to 60
```

Exercise: Increment each element by 10



scores[0] scores[1] scores[2]

```
int scores[]={20,10,50}; // declare an initialize  
//Increment each element by 10
```

C++ 11 range based for loop



scores[0] scores[1] scores[2]

```
int scores[]={20,10,50}; // declare an initialize  
//Print each element using a range based for loop
```

Most common array pitfall- out of bound access



scores[0] scores[1] scores[2]

```
int arr[]={20,10,50}; // declare and initialize
for(int i=0; i<=3; i++)
    scores[i] = scores[i]+10;
```

Tracing code involving arrays



```
int arr[]={1,2,3};  
int tmp = arr[0];  
arr[0] = arr[2];  
arr[2] = tmp;
```

Choose the resulting array after the code is executed

- A.**

1	2	3
arr[0]	arr[1]	arr[2]
- B.**

2	1	3
arr[0]	arr[1]	arr[2]
- C.**

3	2	1
arr[0]	arr[1]	arr[2]
- D.** None of the above

Arrays – motivating example

DEMO: Write a program to store 10 scores and calculate the average of the 10 scores.

Next time

- Pointers
- Mechanics of function calls – call by value and call by reference