PROGRAMMING

CT103

Week 5a

Lecture Content

- Last lecture (Week 4b):
  - Arrays
  - Arrays and loops
  - Arrays example C program
- Today’s morning lecture (Week 5a):
  - Arrays recap
  - Arrays in memory
  - 2D Arrays
  - Example 2D arrays C program

ARRAYS RECAP

Definitions

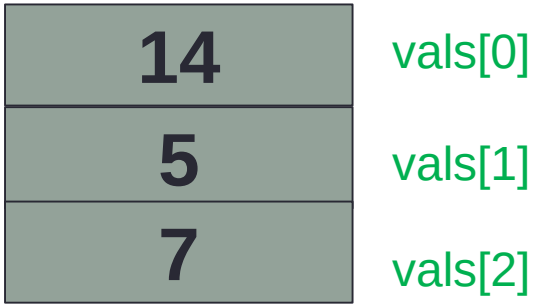
- **Definition**: An **array** is a data structure consisting of a collection of elements. Each element can be identified by an index.

- **Element** – one of the "*items*" in the array.

- **Index** – the position of the element in the array.

Arrays in C

- You can initialise like this:
  - int vals[3] = {14,5,7};



Declaring an Array

- Very straightforward – you just need to specify variable (array) type, name and size, e.g.:
  - `int grades[5];`
- To initialise, you can do like so:
  - `int grades[5] = { 44, 55, 66, 33, 88 };`
- You can implicitly dictate the size of the array:
  - `int grades[] = { 44, 55, 66, 33, 88 }; // size = 5`

Simple Array Problem Example

- The following program creates at array for grade letters.

```c
#include <stdio.h>
void main()
{
    char gradeLetters[] = {'A','B','C','D','F'};
    printf("Grade at index %d is %c.\n",2, gradeLetters[2]);
}
```

Microsoft Visual Studio Debug Console

```
Grade at index 2 is C.
```

Cinema Problem

- This code from last week will read in daily cinema visitors, calculate the average and return the days < average.

```c
#include <stdio.h>
void main()
{
    int visitors[7];
    int sumVisit = 0;
    int avgVisit = 0;

    for (int i = 0; i < 7;i++) {
        printf("Enter visitors for day %d: ",i+1);
        scanf_s("%d",&visitors[i]);
        sumVisit = sumVisit + visitors[i];
    }
    avgVisit = sumVisit / 7;
    printf("Average daily cinema visitors is %d. \n", avgVisit);

    for (int i = 0; i < 7; i++) {
        if(visitors[i]< avgVisit){
            printf("Day %d visitors = %d\n", i+1, visitors[i]);
        }
    }
}
```

Microsoft Visual Studio Debug Console

```
Enter visitors for day 1: 512
Enter visitors for day 2: 523
Enter visitors for day 3: 854
Enter visitors for day 4: 596
Enter visitors for day 5: 1287
Enter visitors for day 6: 1377
Enter visitors for day 7: 1130
Average daily cinema visitors is 897.
Day 1 visitors = 512
Day 2 visitors = 523
Day 3 visitors = 854
Day 4 visitors = 596
```

ARRAYS IN MEMORY

Where/how are arrays stored?

- An array is normally stored in sequential blocks of memory.
- Block size depends on the number of bytes required to store that type of variable.
- For example, an integer usually requires 4 bytes.

Where/how are arrays stored?

- An array is normally stored in sequential blocks of memory.

	grades[0]	grades[1]	grades[2]	grades[3]	grades[4]
	44	55	66	33	88
address:	75F7CC	75F7D0	75F7D4	75F7D8	75F7DC
(this will change every time you run it)	← 4 bytes →	← 4 bytes →	← 4 bytes →	← 4 bytes →	← 4 bytes →

- Functions like *scanf()* need the address of a variable so that it can store new values there
- This is why you put & in front of the variable name, which gives *scanf()* the variable address rather than the variable's current value

Try this out

```
void main()
{
  int grades[5] = { 44, 55, 66, 33, 88 };

  for (int i = 0; i < 5; i++){
    printf("%d stored at address: %X \n", grades[i], &grades[i]);
  }

  printf("\n\n");
}
```

Select Microsoft Visual Studio Debug Co

```
44 stored at address: 75F7CC
55 stored at address: 75F7D0
66 stored at address: 75F7D4
33 stored at address: 75F7D8
88 stored at address: 75F7DC
```

Microsoft Visual Studio Debug Cons

```
44 stored at address: 26FE20
55 stored at address: 26FE24
66 stored at address: 26FE28
33 stored at address: 26FE2C
88 stored at address: 26FE30
```

Copy an array into another

- Easy to do – just use the same index for the source array and the target array. Try this out:

```
#include <stdio.h>

void main()
{
  int grades[5] = { 44, 55, 66, 33, 88 };
  int marks[5];

  for (int i = 0; i < 5; i++){
    marks[i] = grades[i];
  }

}
```

Create an array based on another array

- Easy to run through an array with a for loop and also set the values of another array of the same size

```c
#include <stdio.h>

void main()
{
  double nums[4] = { 1.3, 4.5, 5.123, 6.7002 };
  double squares[4];

  for (int i = 0; i < 4; i++){
    squares[i] = nums[i] * nums[i];
    printf("square of %.2lf = %.2lf \n", nums[i], squares[i]);
  }
}
```

Microsoft Visual Studio Debug Cor

```
square of 1.30 = 1.69
square of 4.50 = 20.25
square of 5.12 = 26.25
square of 6.70 = 44.89
```

2D ARRAYS

2 Dimensional Arrays

- Up until now, we have only considered a 1 dimensional (1D) array.
  - E.g. `int vals[3] = {14,5,7};`

- What if we have 2 dimensional (2D) data that we need to use in our program?

- We use 2D arrays!

2 Dimensional Arrays

- What do 2D arrays look like?
- The following will create a 2-dimensional array of integers:
- int var[2][2];

var[0][0]	var[0][1]
var[1][0]	var[1][1]

- The **first index** is the **row** number, the **second index** is the **column** number.

Initialise 2D array

- Each row is an individual 1D array

- int var[2][2] = {{11,12},{21,22}};

11	12
21	22

Change element

- How do I change an element in a 2D array?

- var[1][0] = 55;

11	12
55	22

Loop over elements in 2D array

- How do I loop over elements in a 2D array?
- You need 2 loops:
  - Outer loop for the rows
  - Inner loop for the columns
  - In the first part of this example, we use two loops to set the values in a 4x4 array. We use a separate variable (val) for the values in the array.

```
int x[4][4];
int r, c, val = 0;

// set array values
for (r = 0; r < 4; r++){
  for (c = 0; c < 4; c++){
    x[r][c] = val;
    val++;
  }
}
```

Output the 2D array

- In the second part of the example we use the same approach to print out the array, using tabs (\t) to space out the values better

```
// output array
for (r = 0; r < 4; r++){
  for (c = 0; c < 4; c++){
    printf("%d\t", x[r][c]);
  }

  printf("\n");
}
```

Microsoft Visual Studio Debug Console

```
0       1       2       3
4       5       6       7
8       9       10      11
12      13      14      15

C:\Users\0063190s\source\repos\tutorial12\
```

Input an array

```c
int x[3][3];
int r, c;

// set array values
for (r = 0; r < 3; r++){
  for (c = 0; c < 3; c++){
    printf("Enter x[%d][%d]: ", r, c);
    scanf_s("%d", &x[r][c]);
  }
}
```

Microsoft Visual Studio Debug C

```
Enter x[0][0]: 11
Enter x[0][1]: 12
Enter x[0][2]: 13
Enter x[1][0]: 21
Enter x[1][1]: 22
Enter x[1][2]: 23
Enter x[2][0]: 31
Enter x[2][1]: 32
Enter x[2][2]: 33
```

And then output the array

```
printf("\n\nThe Array:\n");

// output array
for (r = 0; r < 3; r++){
  for (c = 0; c < 3; c++){
    printf("%d\t", x[r][c]);
  }
  printf("\n");
}
```

```
The Array:
11      12      13
21      22      23
31      32      33
```

EXAMPLE PROBLEMS

Grades Processing Problem

- You are writing software to process student grades for a small class with 5 students. Write a program that:
  - Reads and stores the **semester 1** grades of a subject for 2019 and 2020 classes. Use a 2D array to store these grades. It should look like the following:

	Students				
	1	2	3	4	5
2019	64	81	57	92	41
2020	52	76	42	90	61

  - Create a similar 2D array to store the grades for **semester 2**. Read in the grades from the user.
  - Create a 3rd 2D array to store the final grade calculated as (semester 1 + semester 2)/2. Print this final 2D array to the screen.

Grades Processing Problem

- Go to C program solution.

Grades Processing Problem

- The following code will work:

```c
#include <stdio.h>
void main()
{
    float sem1[2][5];
    float sem2[2][5];
    float finalMark[2][5];
    printf("Semester 1:\n");
    for (int i = 0; i < 2;i++) {
        for (int j = 0; j < 5; j++) {
            printf("Enter semester 1 mark for student %d in year %d: ",j+1,i+2019);
            scanf_s("%f", &sem1[i][j]);
        }
    }

    printf("Semester 2:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 5; j++) {
            printf("Enter semester 2 mark for student %d in year %d: ", j+1, i + 2019);
            scanf_s("%f", &sem2[i][j]);
        }
    }
```

…continue

…continue

```c
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 5; j++) {
            finalMark[i][j] = (sem1[i][j] + sem2[i][j])/2;
        }
    }

    printf("Final marks:\n");
    for (int i = 0; i < 2; i++) {
        printf("\n%d\t",i+2019);
        for (int j = 0; j < 5; j++) {
            printf("%0.2f\t", finalMark[i][j]);
        }
    }

}
```

Grades Processing Problem

- C Program Output:

```
Semester 1:
Enter semester 1 mark for student 1 in year 2019: 56
Enter semester 1 mark for student 2 in year 2019: 95
Enter semester 1 mark for student 3 in year 2019: 85
Enter semester 1 mark for student 4 in year 2019: 45
Enter semester 1 mark for student 5 in year 2019: 65
Enter semester 1 mark for student 1 in year 2020: 85
Enter semester 1 mark for student 2 in year 2020: 91
Enter semester 1 mark for student 3 in year 2020: 75
Enter semester 1 mark for student 4 in year 2020: 68
Enter semester 1 mark for student 5 in year 2020: 95
Semester 2:
Enter semester 2 mark for student 1 in year 2019: 75
Enter semester 2 mark for student 2 in year 2019: 84
Enter semester 2 mark for student 3 in year 2019: 56
Enter semester 2 mark for student 4 in year 2019: 86
Enter semester 2 mark for student 5 in year 2019: 96
Enter semester 2 mark for student 1 in year 2020: 45
Enter semester 2 mark for student 2 in year 2020: 55
Enter semester 2 mark for student 3 in year 2020: 75
Enter semester 2 mark for student 4 in year 2020: 85
Enter semester 2 mark for student 5 in year 2020: 65
Final marks:

2019    65.50   89.50   70.50   65.50   80.50
2020    65.00   73.00   75.00   76.50   80.00
```

Grades Processing Problem

- The previous solution had 4 for loops, can we make our program shorter?
- Yes!
- This will produce the same output.
- Could we make our code shorter again?

```c
#include <stdio.h>
void main()
{
    float sem1[2][5];
    float sem2[2][5];
    float finalMark[2][5];
    printf("Semester 1:\n");
    for (int i = 0; i < 2;i++) {
        for (int j = 0; j < 5; j++) {
            printf("Enter semester 1 mark for student %d in year %d: ",j+1,i+2019);
            scanf_s("%f", &sem1[i][j]);
        }
    }

    printf("Semester 2:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 5; j++) {
            printf("Enter semester 2 mark for student %d in year %d: ", j+1, i + 2019);
            scanf_s("%f", &sem2[i][j]);
            finalMark[i][j] = (sem1[i][j] + sem2[i][j]) / 2;
        }
    }

    printf("Final marks:\n");
    for (int i = 0; i < 2; i++) {
        printf("\n%d\t",i+2019);
        for (int j = 0; j < 5; j++) {
            printf("%0.2f\t", finalMark[i][j]);
        }
    }
}
```