

EECS 10: Computational Methods in Electrical and Computer Engineering

Lecture 11

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 11: Overview

- Data Structures
 - Introduction
 - Arrays
 - Introduction
 - Indexing
 - Initialization
 - Multi-dimensional arrays
 - Operator associativity and precedence
 - Example 1
 - `Dice2.c`
 - Example 2
 - `Histogram.c`

Data Structures

- Introduction
 - Until now, we have used (mostly) single data elements of basic (non-composite) type
 - integral types
 - floating point types
 - Most programs, however, require complex *data structures* using composite types
 - arrays, lists, queues, stacks
 - trees, graphs
 - dictionaries
 - ANSI C provides built-in support for
 - arrays
 - structures, unions, enumerators
 - pointers

Arrays

- Array data type in C
 - Composite data type
 - Type is an array of a sub-type (e.g. array of `int`)
 - Fixed number of elements
 - Array size is fixed at time of definition (e.g. 100 elements)
 - Element access by index (aka. subscript)
 - Element-access operator: `array[index]` (e.g. `A[42]`)
- Example:

```
int A[10]; /* array of ten integers */

A[0] = 42; /* access to elements */
A[1] = 100;
A[2] = A[0] + 5 * A[1];
```

Arrays

- Array Indexing
 - Start counting from 0
 - First element has index 0
 - Last element has index *Size*-1
- Example:

```
int A[10];

A[0] = 42;
A[1] = 100;
A[2] = A[0] + 5 * A[1];
A[3] = -1;
A[4] = 44;
A[5] = 55;
/* ... */
A[9] = 99;
```

| | A |
|---|-----|
| 0 | 42 |
| 1 | 100 |
| 2 | 542 |
| 3 | -1 |
| 4 | 44 |
| 5 | 55 |
| 6 | 0 |
| 7 | 0 |
| 8 | 0 |
| 9 | 99 |

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

5

Arrays

- Array Indexing
 - for loops are often very helpful
 - `for(i=0; i<N; i++)`
`{...A[i]...}`
- Example:

```
int A[10];
int i;

for(i=0; i<10; i++)
{ A[i] = i*10 + i;
}
for(i=0; i<10; i++)
{ printf("%d, ", A[i]);
}
```

| | A |
|---|----|
| 0 | 0 |
| 1 | 11 |
| 2 | 22 |
| 3 | 33 |
| 4 | 44 |
| 5 | 55 |
| 6 | 66 |
| 7 | 77 |
| 8 | 88 |
| 9 | 99 |

```
0, 11, 22, 33, 44, 55, 66, 77, 88, 99,
```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

6

Arrays

- Array Indexing
 - Array indices are *not* checked by the compiler, nor at runtime!
 - Accessing an array with an *index out of range* results in undefined behavior!
- Example:


```
int A[10];
int i;

A[-1] = 42; /* INVALID ACCESS! */

for(i=0; i<=10; i++)
  /* INVALID LOOP RANGE! */
  { printf("%d, ", A[i]);
  }
```

| | |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 0 |
| 8 | 0 |
| 9 | 0 |

EECS10: Computational Methods in ECE, Lecture 11 (c) 2019 R. Doemer 7

Arrays

- Array Initialization
 - Static initialization at time of array definition
 - Initial elements listed in { }
- Example:


```
int A[10] = { 42, 100,
              310, 44,
              55, 0,
              3, 4,
              0, 99};
```

| | A |
|---|-----|
| 0 | 42 |
| 1 | 100 |
| 2 | 310 |
| 3 | 44 |
| 4 | 55 |
| 5 | 0 |
| 6 | 3 |
| 7 | 4 |
| 8 | 0 |
| 9 | 99 |

EECS10: Computational Methods in ECE, Lecture 11 (c) 2019 R. Doemer 8

Arrays

- Array Initialization
 - Static initialization at time of array definition
 - Initial elements listed in { }
- Example:

```
int A[ ] = { 42, 100,
            310, 44,
            55, 0,
            3, 4,
            0, 99};
```

- With given initializer list, array size may be omitted
 - automatically determined

| | A |
|---|-----|
| 0 | 42 |
| 1 | 100 |
| 2 | 310 |
| 3 | 44 |
| 4 | 55 |
| 5 | 0 |
| 6 | 3 |
| 7 | 4 |
| 8 | 0 |
| 9 | 99 |

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

9

Arrays

- Array Initialization
 - Static initialization at time of array definition
 - Initial elements listed in { }
- Example:

```
int A[10] = { 1, 2, 3};
```

- With given initializer list *and* array size, unlisted elements are zero-initialized
 - array is filled up with zeros

| | A |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 3 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 0 |
| 8 | 0 |
| 9 | 0 |

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

10

Arrays

- Multi-dimensional Arrays
 - Array of an array...
- Example:

```
int M[3][2] = {{1, 2},
               {3, 4},
               {5, 6}};

int i, j;

for(i=0; i<3; i++)
  { for(j=0; j<2; j++)
    { printf("%d ",
             M[i][j]);
      }
    printf("\n");
  }
```

| M | 0 | 1 |
|---|---|---|
| 0 | 1 | 2 |
| 1 | 3 | 4 |
| 2 | 5 | 6 |

```
1 2
3 4
5 6
```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

11

Arrays

- Operator associativity and precedence
 - parentheses, **array access** (), [] left to right
 - unary operators +, -, !, ++, -- right to left
 - type casting (*typename*) right to left
 - multiplication, division, modulo *, /, % left to right
 - addition, subtraction +, - left to right
 - shift left, shift right <<, >> left to right
 - relational operators <, <=, >=, > left to right
 - equality ==, != left to right
 - logical and && left to right
 - logical or || left to right
 - conditional operator ?: left to right
 - assignment operators =, +=, *=, etc. right to left
 - comma operator , left to right

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

12

Arrays

- Review program example: `Dice.c` (part 1/4)

```
/* Dice.c: roll the dice */
/* author: Rainer Doemer */
/* modifications: */
/* 10/28/04 RD initial version */

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

/* function definition */

int roll(void)
{
    int r;

    r = rand() % 6 + 1;
    /* printf("Rolled a %d.\n", r); */
    return r;
} /* end of roll */
...
```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

13

Arrays

- Review program example: `Dice.c` (part 2/4)

```
...
/* main function */

int main(void)
{
    /* variable definitions */
    int i, n;
    int count1 = 0, count2 = 0, count3 = 0,
        count4 = 0, count5 = 0, count6 = 0;

    /* random number generator initialization */
    srand(time(0));

    /* input section */
    printf("Roll the dice: How many times? ");
    scanf("%d", &n);

    ...
}
```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

14

Arrays

- Review program example: `Dice.c` (part 3/4)

```

... /* computation section */
for(i = 0; i < n; i++)
{ switch(roll())
  { case 1:
    { count1++; break; }
    case 2:
    { count2++; break; }
    case 3:
    { count3++; break; }
    case 4:
    { count4++; break; }
    case 5:
    { count5++; break; }
    case 6:
    { count6++; break; }
    default:
    { printf("INVALID ROLL!");
      exit(10); }
  } /* hctiws */
} /* rof */
...

```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

15

Arrays

- Review program example: `Dice.c` (part 4/4)

```

...

/* output section */
printf("Rolled a 1 %5d times.\n", count1);
printf("Rolled a 2 %5d times.\n", count2);
printf("Rolled a 3 %5d times.\n", count3);
printf("Rolled a 4 %5d times.\n", count4);
printf("Rolled a 5 %5d times.\n", count5);
printf("Rolled a 6 %5d times.\n", count6);

/* exit */
return 0;
} /* end of main */

/* EOF */

```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

16

Arrays

- Improved program example: `Dice2.c` (part 1/3)

```

/* Dice2.c: roll the dice */
/* author: Rainer Doemer */
/* modifications: */
/* 11/04/04 RD version using arrays */
/* 10/28/04 RD initial version */

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

/* function definition */

int roll(void)
{
    int r;

    r = rand() % 6 + 1;
    /* printf("Rolled a %d.\n", r); */
    return r;
} /* end of roll */

...

```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

17

Arrays

- Improved program example: `Dice2.c` (part 2/3)

```

...
/* main function */

int main(void)
{
    /* variable definitions */
    int i, n;
    int count[6] = { 0, 0, 0, 0, 0, 0 };

    /* random number generator initialization */
    srand(time(0));

    /* input section */
    printf("Roll the dice: How many times? ");
    scanf("%d", &n);

    ...

```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

18

Arrays

- Improved program example: `Dice2.c` (part 3/3)

```

...
/* computation section */
for(i = 0; i < n; i++)
  { count[roll()-1]++;
  } /* rof */

/* output section */
for(i = 0; i < 6; i++)
  { printf("Rolled a %d %5d times.\n",
          i+1, count[i]);
  } /* rof */

/* exit */
return 0;
} /* end of main */

/* EOF */

```

Arrays

- Example session: `Dice2.c`

```

% vi Dice2.c
% gcc Dice2.c -o Dice2 -Wall -ansi
% ./Dice2
Roll the dice: How many times? 6000
Rolled a 1 1009 times.
Rolled a 2 1005 times.
Rolled a 3 962 times.
Rolled a 4 998 times.
Rolled a 5 996 times.
Rolled a 6 1030 times.
% ./Dice2
Roll the dice: How many times? 6000
Rolled a 1 1042 times.
Rolled a 2 983 times.
Rolled a 3 972 times.
Rolled a 4 979 times.
Rolled a 5 1022 times.
Rolled a 6 1002 times.
%

```

Arrays

- Program example: `Histogram.c`
 - Display a simple bar chart for 10 integer values
- Desired output:

```
% ./Histogram
Please enter data value 1: 111
Please enter data value 2: 222
Please enter data value 3: 33
Please enter data value 4: 333
[...]
Please enter data value 10: 111
1: 111 *****
2: 222 *****
3: 33 ****
4: 333 *****
[...]
10: 111 *****
%
```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

21

Arrays

- Program example: `Histogram.c` (part 1/3)

```
/* Histogram.c: print a histogram of data values */
/* author: Rainer Doemer */
/* modifications: */
/* 11/02/04 RD initial version */

#include <stdio.h>

/* constants */
#define NUM_ROWS 10

/* main function */
int main(void)
{
    /* variable definitions */
    int Data[NUM_ROWS];
    int i, j, max;
    double scale;

    ...
}
```

EECS10: Computational Methods in ECE, Lecture 11

(c) 2019 R. Doemer

22

Arrays

- Program example: `Histogram.c` (part 2/3)

```

...
/* input section */
for(i = 0; i < NUM_ROWS; i++)
  { printf("Please enter data value %2d: ", i+1);
    scanf("%d", &Data[i]);
  } /* rof */

/* computation section */
max = 0;
for(i = 0; i < NUM_ROWS; i++)
  { if (Data[i] > max)
    { max = Data[i];
      } /* fi */
  } /* rof */
scale = 70.0 / max;
...

```

Arrays

- Program example: `Histogram.c` (part 3/3)

```

...
/* output section */
for(i = 0; i < NUM_ROWS; i++)
  { printf("%2d: %5d ", i+1, Data[i]);
    for(j = 0; j < Data[i]*scale; j++)
      { printf("");
        } /* rof */
    printf("\n");
  } /* rof */

/* exit */
return 0;
} /* end of main */

/* EOF */

```

Arrays

- Example session: `Histogram.c`

```
% vi Histogram.c
% gcc Histogram.c -o Histogram -Wall -ansi
% ./Histogram
Please enter data value 1: 11
Please enter data value 2: 22
Please enter data value 3: 3
Please enter data value 4: 33
Please enter data value 5: 44
Please enter data value 6: 55
Please enter data value 7: 66
Please enter data value 8: 33
Please enter data value 9: 22
Please enter data value 10: 22
1: 11 *****
2: 22 *****
3: 3 ****
4: 33 *****
5: 44 *****
6: 55 *****
7: 66 *****
8: 33 *****
9: 22 *****
10: 22 *****
%
```

EE