CS2211b

Software Tools and Systems Programming

Western UNIVERSITY · CANADA

Week 10a
Arrays Part 2

Arrays
Part 2

Western Science

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');

        return 0;
}
```

Character Array Example

As characters are treated as integers in C, they can be used as the index in array subscripts.

`a['L']` for example would be equivalent to `a[76]`

Example:

Input characters one at a time using `getchar` until a line break is entered. Count the occurrences of each letter input.

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input
        do {
                ch =

                if(ch
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

Declare an array of 26 characters named `letters`.

Initialize the array to all 0s.

letters:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

Keep looping until `getchar` returns a line break ('\n').

letters:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

Read the next character in the input buffer into the variable ch.

letters:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
```

If the character is a lower case letter then increment the array element at index `ch - 'a'` by one.

This shows the values of the characters such that:

a = 0

b = 1

…

z = 25

and makes them valid array indexes for our letters array.

le

0 … 0

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
```

If the character is an upper case letter, increment the array element at `ch - 'A'` by one.

This shifts the letters in a similar way, such that A = 0, B= 1, … Z = 25. Both 'a' and 'A' will map to 0, 'b' and 'B' to 1, and so on.

```c
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

letters:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch

        printf("Le
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

Print the uppercase letters A to Z in one row.

letters:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = '
                p
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

Print out the counts in the letters array in order on one row.

letters:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

/cs2211/week10/ex1.c

Input Buffer:

R	u	b	b	e	r		D	u	c	k	s	\n

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

letters:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

ys

/cs2211/week10/ex1.c

Input Buffer:

R	u	b	b	e	r		D	u	c	k	s	\n

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

ch = 'R'

ch - 'A'
= 'R' - 'A'
= 82 - 65
= 17

letters:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

Input Buffer:

u	b	b	e	r		D	u	c	k	s	\n

ch = 'u'

ch - 'a'
= 'u' - 'a'
= 117 - 97
= 20

letters:

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>
int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

Input Buffer:

b	b	e	r		D	u	c	k	s	\n

ch = 'b'

ch - 'a'
= 'b' - 'a'
= 98 - 97
= 1

letters:

0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

Input Buffer:

b	e	r		D	u	c	k	s	\n

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
```

ch = 'b'

ch - 'a'
= 'b' - 'a'
= 98 - 97
= 1

letters:

0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

Input Buffer:

e	r		D	u	c	k	s	\n

ch = 'e'

ch - 'a'
= 'e' - 'a'
= 101 - 97
= 4

letters:

0	2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

Input Buffer:

r		D	u	c	k	s	\n

ch = 'r'

ch - 'a'
= 'r' - 'a'
= 114 - 97
= 17

letters:

0	2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

Input Buffer:

	D	u	c	k	s	\n

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

ch = ' '

Character is ignored as it is not a letter. Not in the range 'a' to 'z' or 'A' to 'Z'.

letters:

0	2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

Input Buffer:

D	u	c	k	s	\n

ch = 'D'

ch - 'A'
= 'D' - 'A'
= 68 - 65
= 3

letters:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1	0	0	0	0	0

/cs2211/week10/ex1.c

Input Buffer:

u	c	k	s	\n

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

ch = 'u'

ch - 'a'
= 'u' - 'a'
= 117 - 97
= 20

letters:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	2	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0

/cs2211/week10/ex1.c

Input Buffer:

c	k	s	\n

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

ch = 'c'

ch - 'a'
= 'c' - 'a'
= 99 - 97
= 2

letters:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0

/cs2211/week10/ex1.c

Input Buffer:

k	s	\n

```c
#include <stdio.h>

int main() {
	char letters[26] = {0};
	int i, ch;

	printf("Input a sentence:\n");
	do {
		ch = getchar();

		if(ch >= 'a' && ch <= 'z')
			letters[ch - 'a']++;
		else if(ch >= 'A' && ch <= 'Z')
			letters[ch - 'A']++;
	} while(ch != '\n');

	printf("Letter counts:\n");
	for(i = 'A'; i <= 'Z'; i++) {
		printf("%3c", i);
	}
	putchar('\n');
	for(i = 0; i < 26; i++) {
		printf("%3d", letters[i]);
	}
	putchar('\n');
}
```

ch = 'k'

ch - 'a'
= 'k' - 'a'
= 107 - 97
= 10

letters:

0	2	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	2	0	0	2	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

Input Buffer:

s	\n

ch = 'k'

ch - 'a'
= 's' - 'a'
= 115 - 97
= 18

letters:

0	2	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	2	1	0	2	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

Input Buffer:

\n

```c
#include <stdio.h>

int main() {
        char letters[26] = {0};
        int i, ch;

        printf("Input a sentence:\n");
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

ch = '\n'

Character is ignored as it is not a letter. Not in the range 'a' to 'z' or 'A' to 'Z'.

'\n' causes while loop to terminate.

letters:

0	2	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	2	1	0	2	0	0	0	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

/cs2211/week10/ex1.c

Input Buffer: Empty

```c
#include <stdio.h>
```

Output:

```
Letter counts:
  A  B  C  D  E  F  G  H  I  J  K  L  M  N  O  P  Q  R  S  T  U  V  W  X  Y  Z
  0  2  1  1  1  0  0  0  0  0  1  0  0  0  0  0  0  2  1  0  2  0  0  0  0  0
```

```c
        do {
                ch = getchar();

                if(ch >= 'a' && ch <= 'z')
                        letters[ch - 'a']++;
                else if(ch >= 'A' && ch <= 'Z')
                        letters[ch - 'A']++;
        } while(ch != '\n');

        printf("Letter counts:\n");
        for(i = 'A'; i <= 'Z'; i++) {
                printf("%3c", i);
        }
        putchar('\n');
        for(i = 0; i < 26; i++) {
                printf("%3d", letters[i]);
        }
        putchar('\n');
}
```

letters:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	2	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0	2	1	0	2	0	0	0	0	0

Multidimensional Arrays

- Arrays in C are not limited to a single dimension and may have any number of dimensions.
- Multidimensional arrays are declared in a similar manner to one dimensional arrays.
- **Examples:**

```
int a[5][9];


float b[4][4];


char b[3][3][3];
```

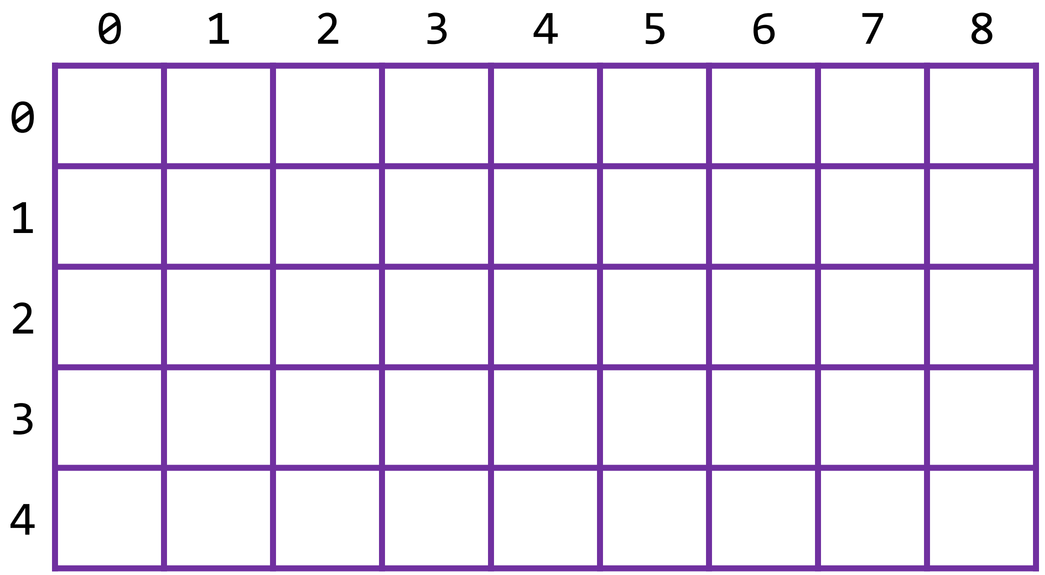
Multidimensional Arrays

- Arrays in C are not limited to a single dimension and may have any number of dimensions.
- Multidimensional arrays are declared in a similar manner to one dimensional arrays.
- **Examples:**

```
int a[5][9];
```

```
float b[4][4];
```

```
char b[3][3][3];
```



Multidimensional Arrays

- Arrays in C are not limited to a single dimension and may have any number of dimensions.
- Multidimensional arrays are declared in a similar manner to one dimensional arrays.
- **Examples:**

```
int a[5][9];

a[2][5] = 4;
float b[4][4];
a[4][7] = -2;


char b[3][3][3];
```

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

Multidimensional Arrays

- Arrays in C are not limited to a single dimension and may have any number of dimensions.
- Multidimensional arrays are declared in a similar manner to one dimensional arrays.
- **Examples:**

```
int a[5][9];
```

```
float b[4][4];

b[0][0] = 3.4f;
b[2][3] = 0.1f;
```

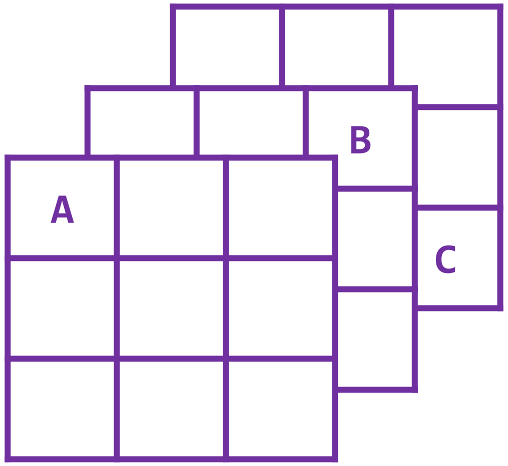
```
char b[3][3][3];
```

	0	1	2	3
0	3.4			
1				
2				0.1
3				

Multidimensional Arrays

- Arrays in C are not limited to a single dimension and may have any number of dimensions.
- Multidimensional arrays are declared in a similar manner to one dimensional arrays.
- **Examples:**

```
int a[5][9];

float b[4][4];

char b[3][3][3];

c[0][0][0] = 'A';

b[1][0][2] = 'B';

b[2][2][2] = 'C';
```



Multidimensional Arrays

- We can initialize multidimensional arrays using similar notation to one dimensional arrays.
- **Example:**

```
int a[5][9] = {{1,2,3,4,5,6,7,8,9},
               {9,8,7,6,5,4,3,2,1},
               {1,1,1,1,1,1,1,1,1},
               {2,2,2,2,2,2,2,2,2},
               {1,0,1,0,1,0,1,0,1}};
```

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

Multidimensional Arrays

- Omitting values (not giving enough values for a row) causes the remaining elements in that row to be set to 0.
- **Example:**

```
int a[5][9] = {{1,2,3,4,5,6,7,8},
               {9,8,7},
               {1,1,1,1,1,1},
               {2},
               {0}};
```

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

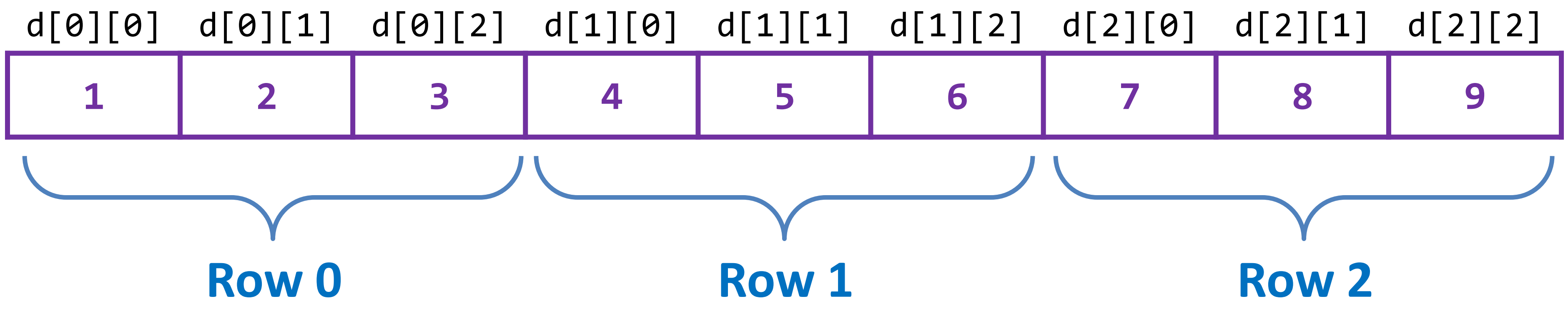
Multidimensional Arrays

- We commonly visualize multidimensional arrays as tables. However, they are actually stored in memory in **row-major order**. That is with row 0 first, then row 2, and so forth in a linear manner.

- **Example:**

```
int d[3][3] = {{1,2,3},
               {4,5,6},
               {7,8,9}};
```

In memory:



Multidimensional Arrays

Size of 2D Arrays

- We can use the `sizeof` operator to find the size of multidimensional arrays like we did with one dimensional arrays.
- `sizeof(a)` gives us the total number of bytes in the array.
- `sizeof(a[0])` gives us the number of bytes in one row of the array.
- `sizeof(a[0][0])` gives us the number of bytes in one element of the array.

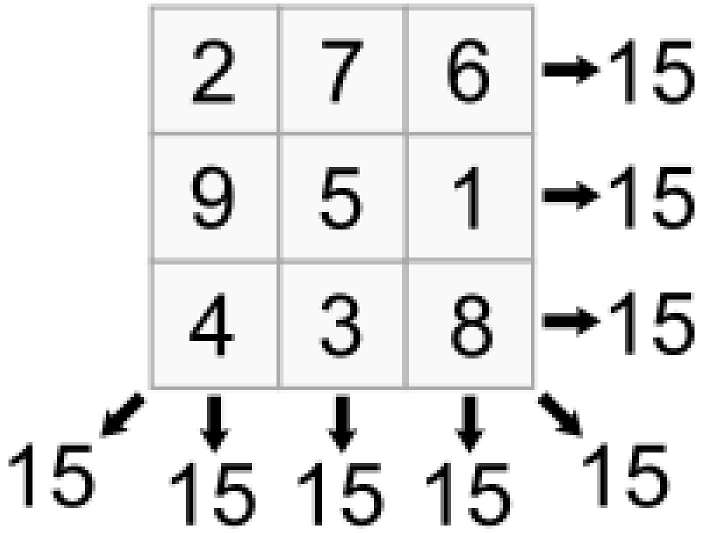
- **Solution:**

```
int a[5][9];

int rows = sizeof(a) / sizeof(a[0]);
int cols = sizeof(a[0]) / sizeof(a[0][0]);
```

Multidimensional Arrays Example

Magic Square

- A magic square is a N x N square grid of positive integers in which each row, column and diagonal has the same sum.
- Write a C program that reads in a magic square of size 3x3 from the user and store it in a multidimensional array.
- Check if the magic square is valid (i.e. that its rows, columns and diagonal sums to the same number).



/cs2211/week10/ex2.c

```
#include <stdio.h>
#define N 3

int main() {
        int square[N][N];
        int r, c, rowsum, colsum, disum = 0;

        // Read in square from user.
        printf("Input %dx%d magic square as %d numbers in order:\n", N, N, N*N);
        for(r = 0; r < N; r++)
                for(c = 0; c < N; c++)
                        scanf("%d", &square[r][c]);

        // Compute the diagonal sum.
        for(r = 0; r < N; r++)
                disum += square[r][r];

        // Compute and check the col and row sums.
        for(r = 0; r < N; r++) {
                rowsum = 0;
                colsum = 0;
                for(c = 0; c < N; c++) {
                        rowsum += square[r][c];
                        colsum += square[c][r];
                }

                if(rowsum != disum || colsum != disum) {
                        printf("Invalid square!\n");
                        return 1;
                }
        }
        printf("Square is valid!\n");
        return 0;
}
```

Arrays as Function Arguments

- Arrays can be passed to functions as arguments.
- When passing one dimensional arrays we **do not need to supply a size**.
- **Example:**

```
int sum_array(int a[]) {
        int i, sum = 0;
        for(i = 0; i < ?; i++)
                sum += a[i];
        return sum;
}
```

How do we get the size of the array?

Arrays as Function Arguments

- Arrays can be passed to functions as arguments.
- When passing one dimensional arrays we **do not need to supply a size**.
- **Example:**

```
int sum_array(int a[]) {
        int i, sum = 0;
        int n = sizeof(a) / sizeof(a[0]);
        for(i = 0; i < n; i++)
                sum += a[i];
        return sum;
}
```

Method we used before will not work when the array is a function parameter.

Arrays as Function Arguments

- Arrays can be passed to functions as arguments.
- When passing one dimen pply **a size**.

Instead we can give the size of the array as a second parameter to the function.

- **Example:**

```
int sum_array(int a[], int n) {
        int i, sum = 0;
        for(i = 0; i < n; i++)
                sum += a[i];
        return sum;
}
```

Arrays as Function Arguments

- **Simple Examples:**

Available as /cs2211/week10/ex3.c

```
void print_array(int a[], int n) {
        int i;

        for(i = 0; i < n; i++)
                printf("%d ", a[i]);

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

```
 int sum_array(int a[], int n) {
        int i, sum = 0;

        for(i = 0; i < n; i++)
                sum += a[i];

        return sum;
 }
```

```
float avg_array(int a[], int n) {
        int sum = sum_array(a, n);
        return (float)sum / n;
}
```

Arrays as Function Arguments

- Unlike simple variables, arrays are passed by reference.
- Changes to the values of the array will affect the original array.

- **Example:**

```
void inc_array(int a[], int n) {
        int i;

        for(i = 0; i < n; i++)
                a[i]++;
}
```

This function will increment all the values in the array by one.

Arrays as Function Arguments

/cs2211/week10/ex3.c

```c
#include <stdio.h>

int sum_array(int a[], int n);
float avg_array(int a[], int n);
void print_array(int a[], int n);
void inc_array(int a[], int n);


int main() {
        int a[5] = {5, 10, 15, -32, 42};

        printf("The array is:\n");
        print_array(a, 5);

        printf("Array sum is: %d\n", sum_array(a, 5));

        printf("The avg is: %.2f\n", avg_array(a, 5));

        printf("Incrementing array.\n");
        inc_array(a, 5);
        print_array(a, 5);

        return 0;
}


…
```

Arrays as Function Arguments

/cs2211/week10/ex3.c

```c
#include <stdio.h>

int sum_array(int a[], int n);
float avg_array(int a[], int n);
void print_array(int a[], int n);
void inc_array(int a[], int n);


int main() {
        int a[5] = {5, 10, 15, -32, 42};

        printf("The array is:\n");
        print_array(a, 5);

        printf("Array sum is: %d\n", sum_array(a, 5));

        printf("The avg is: %.2f\n", avg_array(a, 5));

        printf("Incrementing array.\n");
        inc_array(a, 5);
        print_array(a, 5);

        return 0;
}


…
```

Output:

```
The array is:
5 10 15 -32 42
Array sum is: 40
The avg is: 8.00
Incrementing array.
6 11 16 -31 43
```

Multidimensional Arrays as Arguments

- Multidimensional arrays can also be passed as arguments to functions but **only the length of the first dimension may be omitted**.

- **Example:**

```
#define COLS 3
int sum_array2d(int a[][COLS], int n) {
        int i, j, sum = 0;

        for(i = 0; i < n; i++)
           for(j = 0; j < COLS; j++)
                sum += a[i][j];

        return sum;
}
```

Arrays as Function Arguments Example

- Write a function that sorts an array of doubles using the bubble sort algorithm.

/cs2211/week10/ex4.c

```c
#include <stdio.h>
void slow_sort(double a[], int n);
void print_array(double a[], int n);

int main() {
        double b[] = {5.4, -23.4, 100.1, -3.0, 123.456, 0.0005};

        printf("Before sort:\n");
        print_array(b, 6);

        slow_sort(b, 6);

        printf("After sort:\n");
        print_array(b, 6);

        return 0;
}

void slow_sort(double a[], int n) {
        int i, j;
        double temp;

        for(i = 0; i < n - 1; i++)
                for(j = 0; j < n - i - 1; j ++)
                        if(a[j] > a[j+1]) {
                                temp = a[j];
                                a[j] = a[j+1];
                                a[j+1] = temp;
                        }
}

…
```