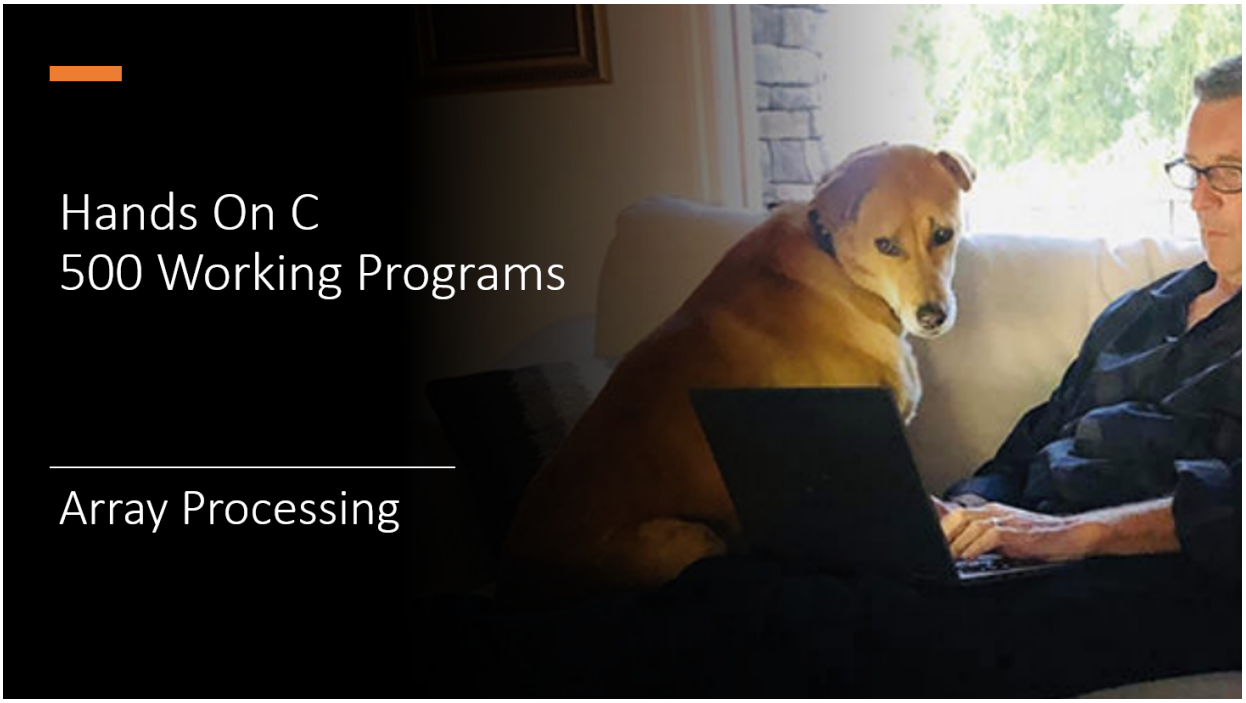

Hands On C
500 Working Programs
Array Processing

studentID[0]	101
studentID[1]	201
studentID[2]	301
studentID[3]	401
studentID[4]	501
studentID[5]	601
studentID[6]	701
studentID[7]	801
studentID[8]	901
studentID[9]	1001

Understanding Arrays

In [1]:
```
#include <stdio.h>

int main(void)
{
    int studentID[10];
    double studentGPAs[10];

    printf("sizeof studentID array: %ld bytes\n", sizeof(studentID));
    printf("sizeof studentGPA array: %ld bytes", sizeof(studentGPAs));
}
```

```
sizeof studentID array: 40 bytes
sizeof studentGPA array: 80 bytes
```

Assigning Values to an Array

In [2]:
```c
#include <stdio.h>

int main(void)
{
   int scores[5];

   scores[0] = 100;  // first score is at index 0
   scores[1] = 97;
   scores[2] = 88;
   scores[3] = 92;
   scores[4] = 85;
}
```

In [3]:
```c
#include <stdio.h>

int main(void)
{
   int scores[5];

   scores[0] = 100;  // first score is at index 0
   scores[1] = 97;
   scores[2] = 88;
   scores[3] = 92;
   scores[4] = 85;

   printf("%d %d %d %d %d\n", scores[0], scores[1], scores[2], scores[3], scores[
}
```

```
100 97 88 92 85
```

In [4]:
```c
#include <stdio.h>

int main(void)
{
   int scores[5];

   scores[0] = 100;  // first score is at index 0
   scores[1] = 97;
   scores[2] = 88;
   scores[3] = 92;
   scores[4] = 85;

   for (int i = 0; i < 5; i++)
     printf("%d ", scores[i]);
}
```

```
100 97 88 92 85
```

In [5]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

int main(void)
{
   int scores[ARRAY_SIZE];

   scores[0] = 100;  // first score is at index 0
   scores[1] = 97;
   scores[2] = 88;
   scores[3] = 92;
   scores[4] = 85;

   for (int i = 0; i < ARRAY_SIZE; i++)
     printf("%d ", scores[i]);
}
```

```
100 97 88 92 85
```

In C, Characters Strings Are Arrays

In [6]:
```c
#include <stdio.h>

#define ARRAY_SIZE 4

int main(void)
{
   char name[ARRAY_SIZE];

   name[0] = 'K';
   name[1] = 'r';
   name[2] = 'i';
   name[3] = 's';

   for (int i = 0; i < ARRAY_SIZE; i++)
     printf("%c", name[i]);
}
```

```
Kris
```

In [7]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

int main(void)
{
   char name[ARRAY_SIZE];

   name[0] = 'K';
   name[1] = 'r';
   name[2] = 'i';
   name[3] = 's';
   name[4] = '\0';  // null character marks the end of a string

   printf("%s", name);
 }
```

```
Kris
```

In [8]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

int main(void)
{
   char name[ARRAY_SIZE];
   int count = 0;

   name[0] = 'K';
   name[1] = 'r';
   name[2] = 'i';
   name[3] = 's';
   name[4] = '\0';  // null character marks the end of a string

   for (int i = 0; name[i] != '\0'; ++i)
     count += 1;

   printf("The string %s contains %d characters", name, count);
 }
```

```
The string Kris contains 4 characters
```

Initializing an Array

In [9]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

int main(void)
{
   int scores[ARRAY_SIZE];

   scores[0] = 100;  // first score is at index 0
   scores[1] = 97;
   scores[2] = 88;
   scores[3] = 92;
   scores[4] = 85;

   for (int i = 0; i < ARRAY_SIZE; i++)
     printf("%d ", scores[i]);
}
```

```
100 97 88 92 85
```

In [10]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

int main(void)
{
   int scores[ARRAY_SIZE] = { 100, 97, 88, 92, 85 };

   for (int i = 0; i < ARRAY_SIZE; i++)
     printf("%d ", scores[i]);
}
```

```
100 97 88 92 85
```

In [11]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

int main(void)
{
   char name[ARRAY_SIZE] = "Kris";

   printf("%s", name);
}
```

```
Kris
```

In [12]:
```c
#include <stdio.h>

int main(void)
{
   int scores[] = { 100, 97, 88, 92, 85, -1 };

   for (int i = 0; scores[i] != -1; i++)
     printf("%d ", scores[i]);
}
```

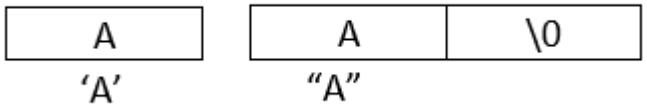
```
100 97 88 92 85
```

Understanding Buffer Overflow

In [13]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

int main(void)
{
   int scores[ARRAY_SIZE] = { 100, 97, 88, 92, 85 };

   for (int i = 0; i < ARRAY_SIZE; i++)
     printf("%d ", scores[i]);

   scores[6] = 100;  // Overflow
}
```

```
100 97 88 92 85

*** stack smashing detected ***: <unknown> terminated
[C kernel] Executable exited with code -6
```

How "A" is Different from 'A'



In [14]:
```c
#include <stdio.h>

int main(void)
{
   char letter = 'A';
   char string[] = "A";

   printf("%c\n", letter);
   printf("%s\n", string);
   printf("sizeof letter = %ld\n", sizeof(letter));
   printf("sizeof string = %ld", sizeof(string));
}
```

```
A
A
sizeof letter = 1
sizeof string = 2
```

Understanding Multidimensional Arrays

studentID[0]	101
studentID[1]	201
studentID[2]	301
studentID[3]	401
studentID[4]	501
studentID[5]	601
studentID[6]	701
studentID[7]	801
studentID[8]	901
studentID[9]	1001

[0][0]	[0][1]	[0][2]
[9]0]	[9][1]	[9][2]

In [15]:
```c
#include <stdio.h>

#define NUMBER_OF_STUDENTS 10
#define NUMBER_OF_TESTS 3

int main(void)
{
  int studentID[NUMBER_OF_STUDENTS];
  int grades[NUMBER_OF_STUDENTS][NUMBER_OF_TESTS];

  printf("sizeof studentID is %ld\n", sizeof(studentID));
  printf("sizeof grades is %ld", sizeof(grades));
}
```

```
sizeof studentID is 40
sizeof grades is 120
```

Initializing a Multidimensional Array

In [16]:
```
#include <stdio.h>

#define NUMBER_OF_STUDENTS 5
#define NUMBER_OF_TESTS 3

int main(void)
{
  int studentID[NUMBER_OF_STUDENTS] = {100, 101, 102, 103, 104 };
  int grades[NUMBER_OF_STUDENTS][NUMBER_OF_TESTS] = {{ 90, 80, 95 },
                                                     { 92, 84, 77 },
                                                     { 88, 73, 58 },
                                                     { 82, 88, 0 },
                                                     { 44, 55, 87 }};
}
```

In [17]:
```
#include <stdio.h>

#define NUMBER_OF_STUDENTS 5
#define NUMBER_OF_TESTS 3

int main(void)
{
  int studentID[NUMBER_OF_STUDENTS] = {100, 101, 102, 103, 104 };
  int grades[NUMBER_OF_STUDENTS][NUMBER_OF_TESTS] = {{ 90, 80, 95 },
                                                     { 92, 84, 77 },
                                                     { 88, 73, 58 },
                                                     { 82, 88, 0 },
                                                     { 44, 55, 87 }};

  printf("Student\tTest Score\n");
  for (int student = 0; student < NUMBER_OF_STUDENTS; ++student)
    {
      printf("%d\t", studentID[student]);
      for (int test = 0; test < NUMBER_OF_TESTS; ++test)
        printf("%d ", grades[student][test]);

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

```
Student Test Score
100     90 80 95
101     92 84 77
102     88 73 58
103     82 88 0
104     44 55 87
```

Too Many Dimensions Become Confusing and Hard to Read

In [18]:
```c
#include <stdio.h>

#define YEARS 3
#define MONTHS 12
#define DAYS 31
#define ORDERS 5
#define CLIENTS 5

int main(void)
{
  int sales[YEARS][CLIENTS][MONTHS][DAYS][ORDERS];

  sales[0][1][10][12][5] = 8;  // 8 orders on 10/12 in first year for client 1
  printf("sizeof sales is %ld bytes\n", sizeof(sales));
}
```

```
sizeof sales is 111600 bytes
```

Passing an Array to a Function

In [19]:
```c
#include <stdio.h>

#define ARRAY_SIZE 5

void show_array(int array[], int size)
{
   for (int i = 0; i < size; ++i)
     printf("%d ", array[i]);
}

int main(void)
{
   int scores[ARRAY_SIZE] = { 100, 97, 88, 92, 85 };

   show_array(scores, ARRAY_SIZE);
}
```

```
100 97 88 92 85
```

In [20]:
```c
#include <stdio.h>

void show_string(char string[])
{
   for (int i = 0; string[i] != '\0'; ++i)
     printf("%c", string[i]);
}

int main(void)
{
   char firstname[] = "Kris";
   char lastname[] = "Jamsa";

   show_string(firstname);
   show_string(lastname);
}
```

```
KrisJamsa
```

In [21]:
```c
#include <stdio.h>

void show_string(char string[])
{
   for (int i = 0; string[i] != '\0'; ++i)
     printf("%c", string[i]);
}

int main(void)
{
   char firstname[] = "Kris";
   char lastname[] = "Jamsa";

   show_string(firstname);
   show_string(" ");
   show_string(lastname);
}
```

```
Kris Jamsa
```

Passing a Multidimensional Array to a Function

In [22]:
```c
#include <stdio.h>

#define NUMBER_OF_STUDENTS 5
#define NUMBER_OF_TESTS 3

void show_scores(int id[], int grades[][NUMBER_OF_TESTS], int rows, int columns)
{
  printf("Student\tTest Score\n");

 for (int student = 0; student < rows; ++student)
    {
      printf("%d\t", id[student]);
      for (int test = 0; test < columns; ++test)
        printf("%d ", grades[student][test]);

      printf("\n");
    }

}

int main(void)
{
  int studentID[NUMBER_OF_STUDENTS] = {100, 101, 102, 103, 104 };
  int grades[NUMBER_OF_STUDENTS][NUMBER_OF_TESTS] = {{ 90, 80, 95 },
                                                     { 92, 84, 77 },
                                                     { 88, 73, 58 },
                                                     { 82, 88, 0 },
                                                     { 44, 55, 87 }};

  show_scores(studentID, grades, NUMBER_OF_STUDENTS, NUMBER_OF_TESTS);
}
```

```
Student Test Score
100     90 80 95
101     92 84 77
102     88 73 58
103     82 88 0
104     44 55 87
```

What You will Learn Next

```
To better manage their code, to improve readability, and to increase the code
reuse, programmers often break a larger program into smaller pieces called
functions. Each function should perform a specific task, such as the printf
function that displays output on the screen.

#include <stdio.h>

void sayHello(void)
{
  printf("Hello, world\n");
}

int main(void)
{
  sayHello();
}
```