structs

Creation and manipulation of records
Goals

- By the end of this lesson you will be able to:
  - Create and use structs in a variety of ways
What are structs?

• Multiple data types may be needed to describe something
  - Example: Data about a student may need to be stored/manipulated in a program. The relevant data may include their name, address, student ID, GPA, etc.

• We can create our own data type called a record, structure, or simply struct to encapsulate all of this data in one unit

• The simpler data types within can be manipulated, output, etc. as necessary
Creating structs

- Like `typedef` and `enum`, before we can create a variable of a user-created data type, we must define it

- Define structs as in this example:
  ```
  struct student
  {
    string name, address;
    double gpa;
    int credit_hours_by_semester[20];
  };
  // note the semicolon!!
  ```

- Typically we place this definition near the top of the program (along with `enum` and `typedef`, before function prototypes)
Creating `struct` variables

- With this definition:
  ```
  struct student
  {
    string name, address;
    double gpa;
    int credit_hours_by_semester[20];
  };
  ```

- Create variables as follows:
  ```
  student s;
  student s1, s2;
  student stuarray[50];
  ```
Accessing struct elements

- With this definition and declaration:

```cpp
struct student
{
    string name, address;
    double gpa;
    int credit_hours_by_semester[20];
};

... student s;
```

- Access elements as follows:

```cpp
s.name = "Joe Smith";
s.gpa = 3.84;
cout << s.gpa << endl;
cout << "GPA squared is: " << pow(s.gpa, 2) << endl;
s.credit_hours_by_semester[3] = 12;
```
Arrays of structs

- Accessing individual elements of an array of structs:
  ```
  student students[20];
  students[0].name = "Bob Davis";
  students[0].gpa = 2.77;
  cout << students[0].gpa << endl;
  ```
As a struct is just a data type, you can nest them:

```c
struct name
{
    string first;
    string last;
};
struct student
{
    name n;
    double gpa;
};
...
student s;
s.n.first = "Charlie";
s.n.last = "Smith";
```
structs and functions

• You can pass structs to a function:
  void fcn(student, int); // value
  void fcn(student &, int); // reference
  void fcn(const student &, int); // by constant reference

• You can return a struct from a function:
  student fcn(string, int);
Exploration

- Think about prior programs you have written where you have stored and manipulated data and how you might rewrite them to use structs instead.

- Design a struct to store information about integers where one piece of data is the integer itself and the remaining data are booleans that describe if the number is even or odd, prime or composite, perfect or not, square or not, triangular or not, etc. Create a large array (say 1,000,000) of these structs, populate their data and then conduct various experiments, for example examine the distribution of primes or find how many triangular numbers are also square over a certain range.