enum and typedef

Creating your own simple datatypes
Goals

- By the end of this lesson you will be able to:
  - Understand and use enumerated types
  - Understand and use type definitions
Enumerated types

- It may be useful to represent data in a way that is not universally general, but would be general in the context of our program.
- For a program involving a school, it may be useful to have a data type that can take on values of freshman, sophomore, junior, and senior.
Enumerated types

- Enumeration is simply a ordered listing of elements, thus each element can be described by both its name and its position.
- In terms of programming, think of it as a mapping.
- Examples of enumeration that are already in C++:
  - `bool` has two values: `false`, `true`
    - `false` maps to 0, `true` maps to 1
  - `char` has many possible values
    - 'A' maps to 65, 'B' maps to 66, etc.
Using `enum`

- `enum` allows you to create your own enumerated type
  ```cpp
  enum typename { value1, value2, ... };
  ```
- `typename` is any valid identifier
  - By convention, we avoid using typenames ending in `_t`
- `value1, value2, ...` are one or more of valid identifiers
  - By convention, these values should be uppercase like constants
- Usually create an `enum` definition towards the top of your program – usually before function prototypes
- Usage is similar to `bool`
Behind the scenes

• What happens when you create an enumerated type?
• The first value you list is assigned a value of 0, the next one is assigned the value 1, etc. thus you can use integers, do math with the values, etc.
• If you output the values, the output will be the integral value.
• Enumerated types are typically only used with respect to assignment, math, and logic (if, loops, switch/case), usually to make code more readable.
Additional details

- You cannot use the same identifier in multiple `enum` definitions.
- You can define the integral value of each enumerated value – successive values increase by one – this means it is possible to have multiple enumerated values with the same integral value.
- When doing certain operations you may need to typecast.
Type definitions

• You can create another name for a datatype using a type definition as follows:
  
  ```
  typedef existing_datatype new_datatype;
  ```

• This can or should be used when it improves the readability and understanding of code

• Like `enum`, this definition typically goes before function prototypes
Exploration

- Think of programs you might write that would benefit from the use of an enumerated type or type definition.
- Write a program using an enumerated type to represent grades (A, B, C, D, and F). Have the user input a numerical grade, then store and output their grade using the enumerated type.