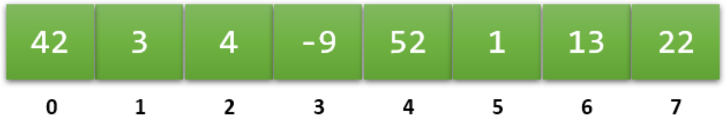
Structured Types

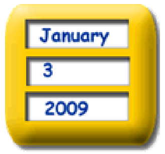
Primitive types are fine for creating simple programs. But, for most tasks, you need more complex **user-defined data types**, such as `string` and `vector`. When we combine multiple data items into a larger unit, it is called a **structured** type. The types in the standard library, such as `string` and `vector`, are **structured** types.

The C++ **language** has two derived structured data types. The built-in linear **list-type** collection is called an **array**. In an array, all of the **elements** must be of the same type, so we say that an array is a **homogeneous** structured type. With an array, the elements are accessed by using their **index**, just like the `string` type.



In addition to arrays, programs often **combine related pieces of information** into a composite object which can be manipulated as a unit, such as an **employee record**. Each worker has an employee number, a name, an address, job title and so on. Such types are called **records** (which is a generic Computer Science term) or **structures** (which is the C/C++ specific term).

The **data members** which make up a structure do **not** all need to be of the same type, so we say that a structure is a **heterogenous** data type. The `Date` type shown here is a **structure** type, consisting of a month, day and year. C++ also has more advanced record types, called **classes**, which you'll study later in the semester.



January
3
2009