

# Chapter 2

## Creating a C++ Program

### Elements of a C++ Program

- Four basic ways of structuring a program
  1. Sequencing
  2. Selection
  3. Looping
  4. Sub-program

## Basics

- All programs must have at least one function
- We will learn to write many functions
- We will build more complex programs from simple functions

## Parts of the Programs

- Identifiers
  - A name associated with a function or data object
  - It is used to reference that function or data object

## How to Name Identifiers

- Must start with a letter or underscore
- Followed by a letter, number or underscores
- Cannot be a reserved word

## Quiz

- Write true if the identifier's name good and false otherwise:

`_Fred`  
`3DArray`  
`R2D2`  
`Silly_Name`  
`Good Name`  
`x`  
`BadName`  
`Id3nt1f13r`  
Bonus:  
`int`  
`const_cast`

## Examples of Reserved Words

- int
- double
- const
- return
- bool
- if
- while
- for

## Data and Data Types

- Data type
  - Determines how the data is represented in the computer and how the computer can operate on it
- Data
  - What all computers use to compute

# Data Types

- Two Kinds
  - Built-in
  - User Defined
- Example of Built-in
  - int
  - double
  - bool
  - char

## char Data Type

- Any single alphanumeric character
  - Example
  - 'A'
  - 'a'
  - 'B'
  - '1'
  - '^'

## string Data Type

- Not a built-in type
- Any sequence of characters
  - Example
  - “Dave” “C++” “ 9 ”
- Strings cannot span more than one line
- Null String
  - “”

## Naming Elements

- Use a Declaration
- `int empNum;`
- Variables
  - A location in memory, referenced by an identifier, that contains a data value that can be changed
- Constants
  - A location in memory, referenced by an identifier, that contains a data value that cannot be changed

## Examples

- `int Counter;`
- `double taxRate, Percent;`
- `const char MIDDLEINITIAL = 'P';`
- `const bool NOTTRUE = "false";`

## Executable Statements

- **Assignment Statements**
  - `lastName = "Roszak";`
- **Output Statements**
  - `cout << "Hello World";`

## Non-executable Statements

- Comments

- //This is how you indicate a comment
- /\*Or you can do it this way\*/

## Blocks

- Groups of lines that are to be grouped together between curly braces

- Example

```
– int main()  
  {  
    cout << "Hello World;  
  }
```

## Pre-processor

- Before the program is compiled, a program called a pre-processor parses the code looking for directives; things to do
- Example
  - #include <iostream>
  - #ifndef
  - #endif

## Introduction to Namespaces

```
int main()
{
    cout << "Happy Birthday" << endl;
    return 0;
}
```

## More Namespaces

```
#include <iostream>
```

```
int main()
```

```
{
```

```
    cout << "Happy Birthday" << endl;
```

```
    return 0;
```

```
}
```

## Still More

```
#include <iostream>
```

```
int main()
```

```
{
```

```
    std::cout << "Happy Birthday" <<
```

```
    std::endl;
```

```
    return 0;
```

```
}
```

## Yet More

```
#include <iostream>
using std::cout;
using std::endl;
int main()
{
    cout << "Happy Birthday" << endl;
    return 0;
}
```

## One More

```
#include <iostream>
using namespace std;
int main()
{
    cout << "Happy Birthday" << endl;
    return 0;
}
```

## More Output

- What will the following lines produce?
  - `cout << "Hi there. " << endl;`
  - `cout << endl;`
  - `cout <<"What are you doing?" << endl;`

## More Output

- What will these produce?
  - `cout << "Hi there. " << endl << endl`  
`<<"What are you doing?" << endl;`
- How about these?
  - `cout <<"Hi there. \n\nWhat are you`  
`doing?\n"`