**Compiler Errors**

* Programming languages have rules;
* Syntax errors – something wrong with the structure;

**Std:: cout << ``Errors << std::endl;**

**return 0 ( is missing a semicolon)**

* Semantic errors – something wrong with the meaning

**a + b**; When it doesn´t make sense to add **a** and **b.**

**Compiler Warnings**

**Do NOT ignore them!**

* The compiler has recognized an issue with your code that could lead to a potential problem;
* It´s only a wraning because the compile ris still able to generate correct machine code;

**int miles\_driven;**

**std::count << miles\_driven; (**this will give us a random number)

warning: ‘miles\_driven’ is used uninitialized …

**Linker Errors**

* The linke ris having trouble linking all the object files together to create na executable;
* Usually there is a library or object file that is missing.
* The error generated when all the parts that make up a program cannot be put together because one or more are missing is called a **Linker error**

**Runtime Errors**

* Errors that occur when the program is executing
* Some typical runtime errors
* Divide by zero;
* File not found;
* Out of memory.
* Can cause the program to ‘crash’
* Exception Handling can help deal with runtime errors.

**Logic Errors**

* Errors or bugs in your code that cause your program to run incorrectly;
* Logic errors are mistakes made by the programmer;
* Suppose we have a program that determines if a person can vote in na election and you must be 18 years or older to vote.

If (age > 18) {

Std::cout << “Yes, you can vote!”;

}

**Compilers**

**Compilers** are used to translate programming language source code instructions into the appropriate machine language instructions.

**Algorithm**

A set of precise steps for solving a problem is known as a(n):