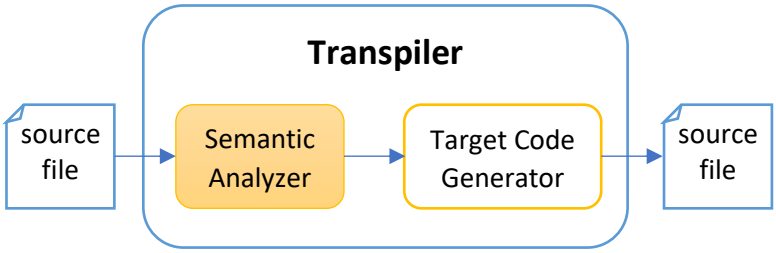
Lab-02 Add a simple semantic analysis

In this lab, we will extend the **transpilers** from lab-01 by adding a simple semantic analyzer. In this lab, the compiler contains only two components:

1. Semantic analyzer: For this lab, the semantic analyzer will perform range checking to ensure that the number is within the appropriate range.
2. Target code generator: For this lab, the target code generator performs the similar tasks as lab-01.



1 Objectives

1. Construct a compiler incrementally in a back-to-front approach.
2. Extend the transpiler with a simple semantic analyzer.

2 Learning Outcomes

By completing this lab, learners should be able to

1. build a simple semantic analyzer.
2. construct a simple compiler with two components:
    a. semantic analyzer
    b. target code generator.

3 Tasks

In this lab, we will check whether the input number is between 0 and 2,147,483,647, i.e. whether it is a positive 32-bit integer. If the input number is out of the specified range, display an error and the transpiler should stop. On the other hands, if the input number is within the specified range, the transpiler should generate the target codes similar to lab-01.

Note that it depends on how you implement the programs in lab-01.

1. You may need to change the codes in lab-01 to ensure that the output source code print the appropriate type. For instance, the C program may need to use the following statement:

```
printf("%d", value);
```

2. You may need to implement 1 or 5 programs, depending on how you structure your transpiler program.