TITTLE:

INTRODUCTION

In an implementation of a compiler, portion of one or more phases are combined into a module called pass.

It is a group of phases implemented together

A pass reads the source program of the output of the previous pass, makes the transformations specified by its phases, and writes output into an intermediate file, which is read by a subsequent pass

In pass, the operations of phases may be interleaved

Generally, the lexical analysis, scanner, semantic analysis and intermediate code generation are grouped into one pass whereas the remaining phases into another pass

Number of passes to keep is generally dependent upon the structure of the source language and the environment in which the compiler must operate

It is better to have less number of passes as it takes time to read and write intermediate files

However it is not always possible to have a single pass instead of several passes, as sometimes there may be a need to keep the entire program in memory because one phase may need information in a different order than a previous phase produces

The most general term for a software code converting tool is “translator.” A translator, in software programming terms, is a generic term that could refer to a compiler, assembler, or interpreter; anything that converts higher level code into another high-level code (e.g., Basic, C++, Fortran, Java) or lower-level (i.e., a language that the processor can understand), such as assembly language or machine code. If you don’t know what the tool actually does other than that it accomplishes some level of code conversion to a specific target language, then you can safely call it a translator.

AIM

In this lab we make a tpascal program that is able to pass and translate general expressions that involve addition and subtraction.We Used the code given below and extended Cradle.pas to provide the correct assembly for translating binary expressions such as "\b 1 + 2" \b0 or "\b 5 - 3". ***\***

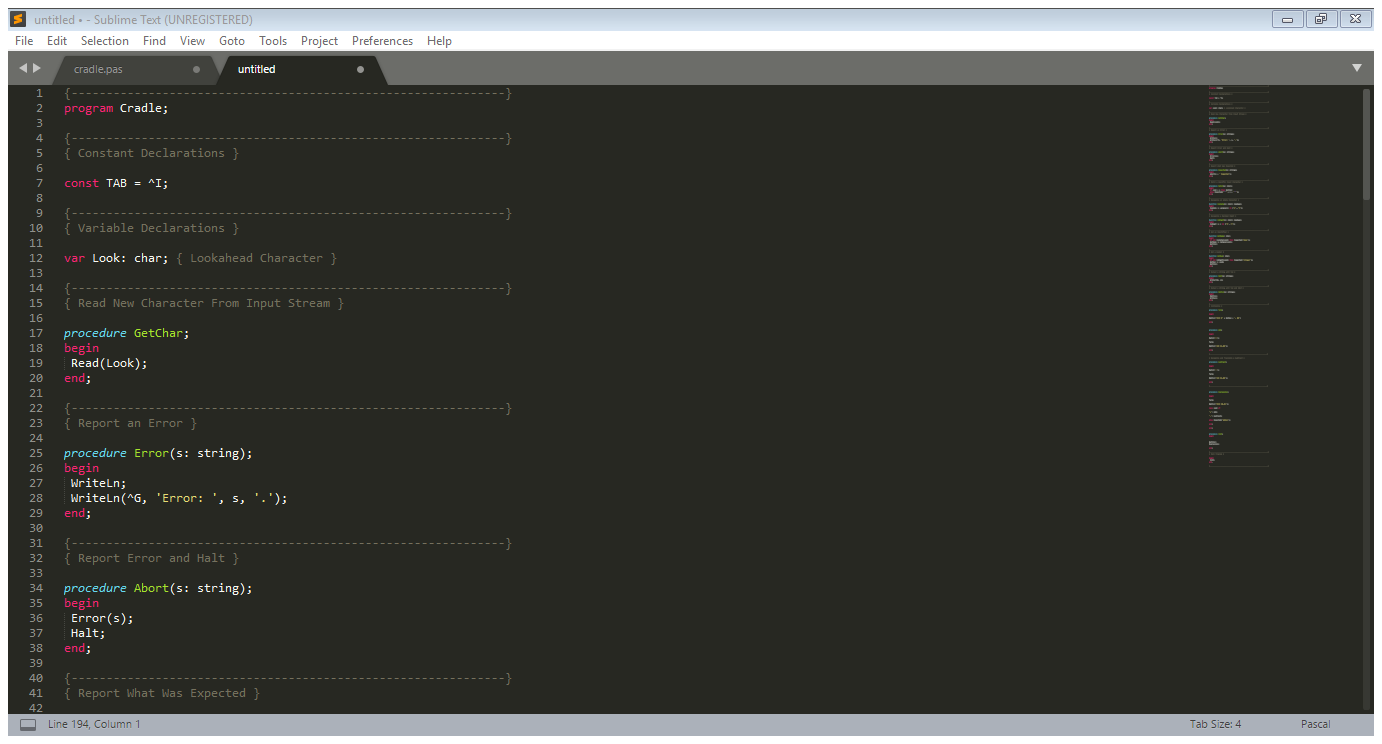


Figure 1

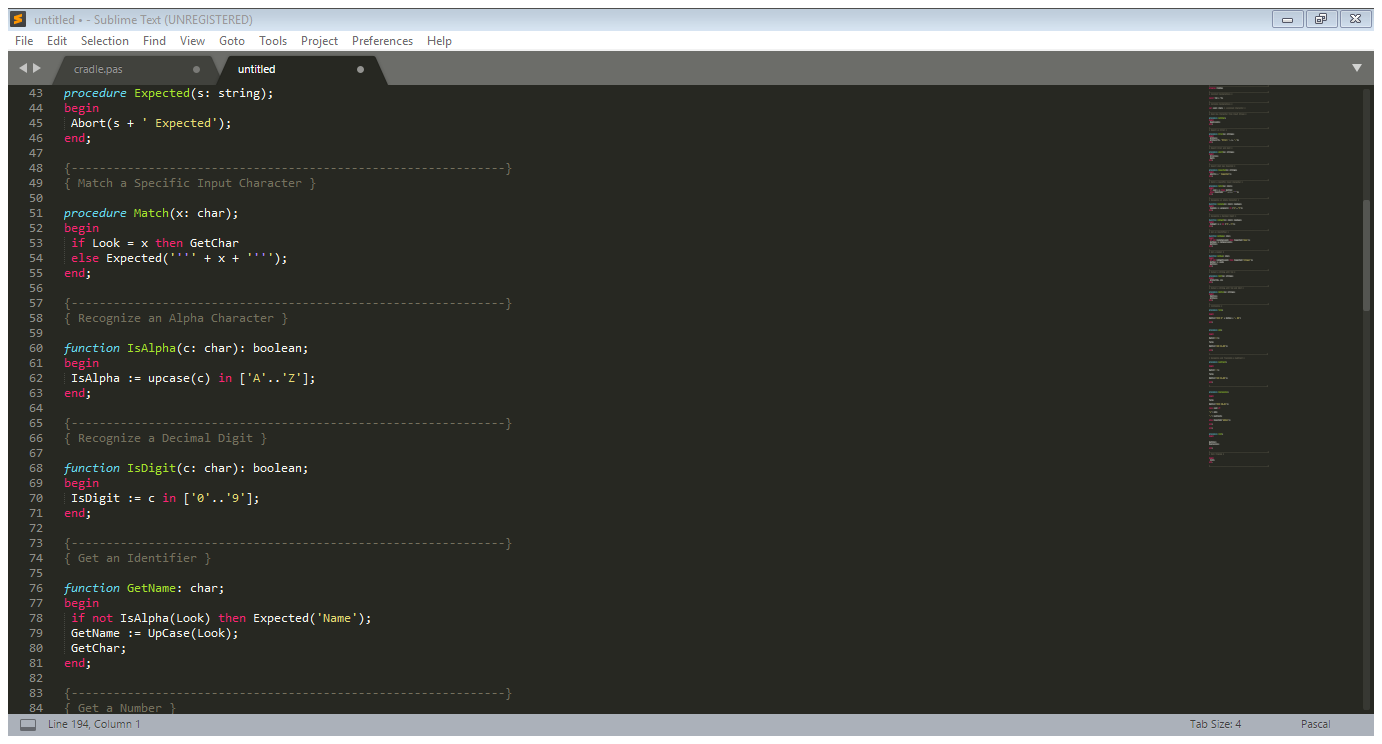


Figure 2

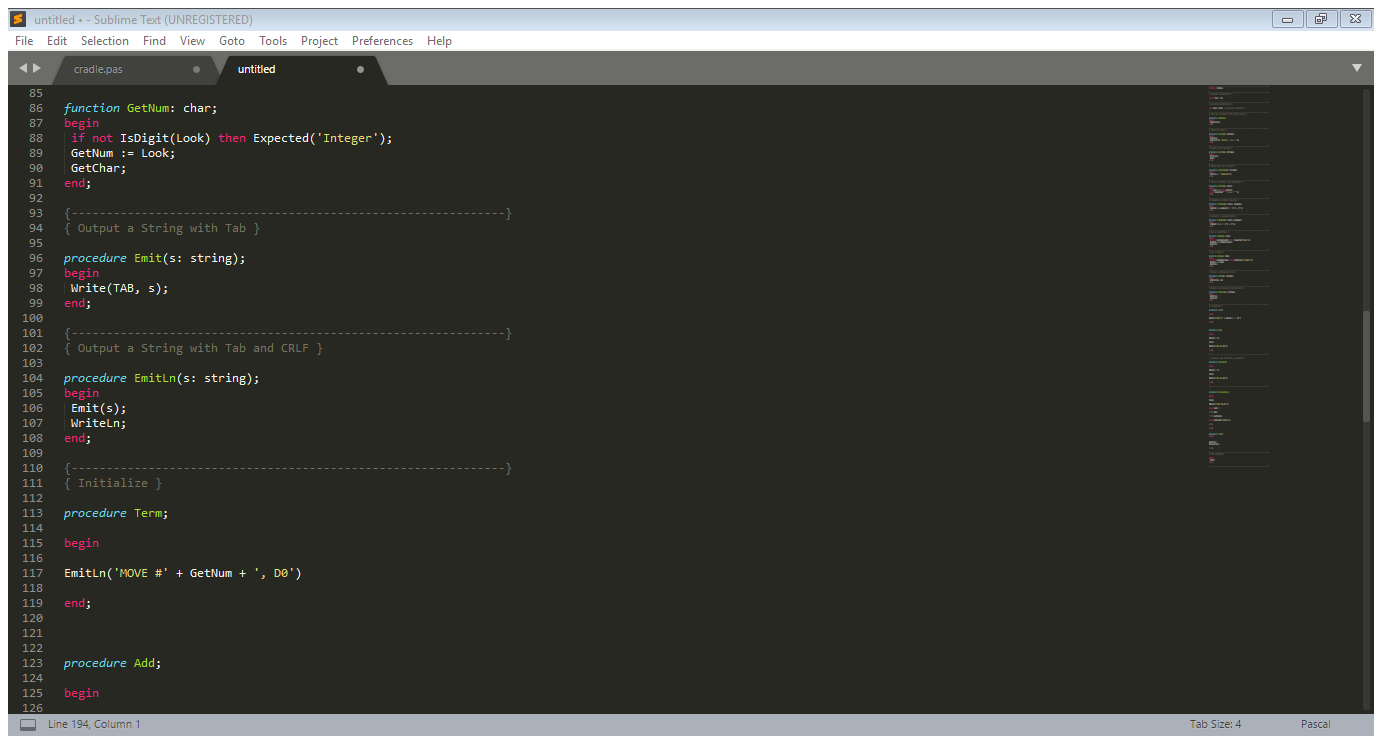


Figure 3

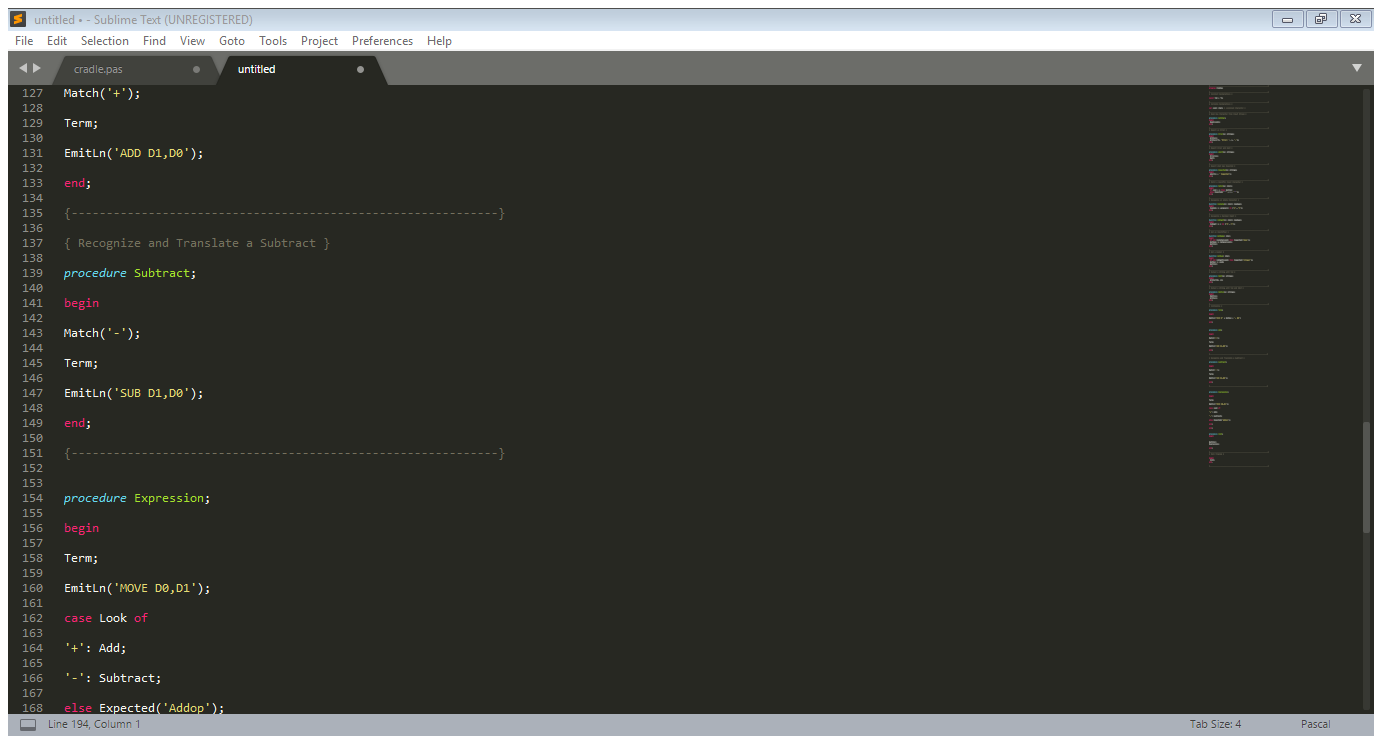


Figure 4

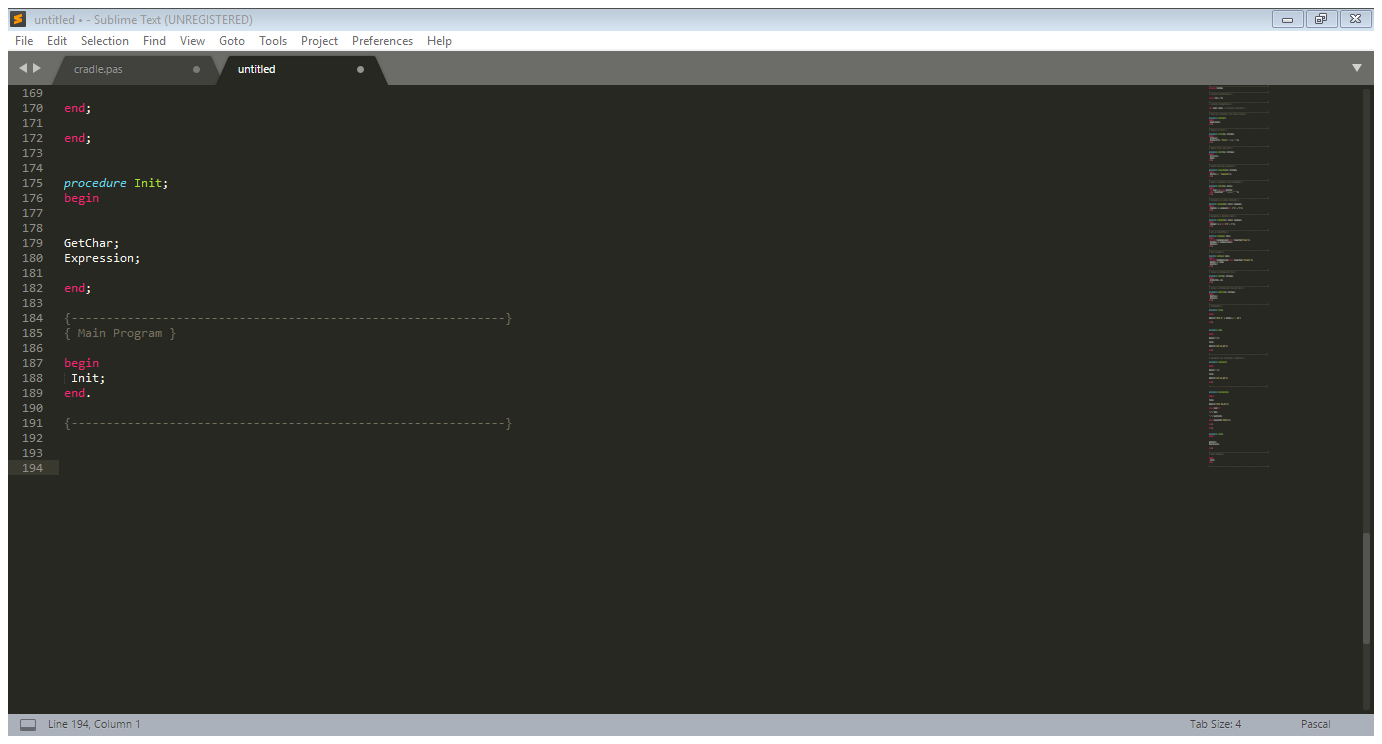


Figure 5

***Figure1 to figure5 shows the program we wrote and compiled it via tpascal4***

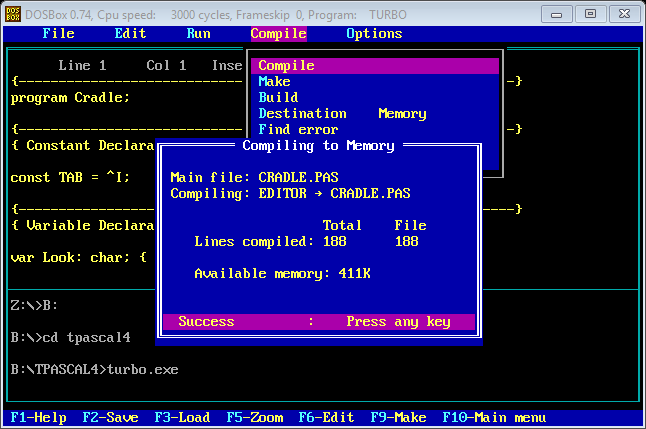


Figure 6

***After writing the program we loaded it to tpascal4 and compiled it, and figure6 shows that the compiling of our program was successful.***

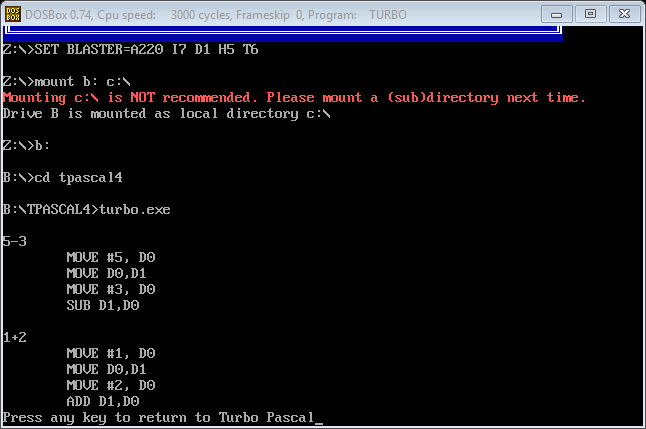


Figure 7

***Figure 7 shows that we ran the program and it showed the results of what was required***

CONCLUSION

Our compiling was successful and our program was able to run.at first we faced some errors on “Term” because we didn’t include the procedure term but after we did our program was able to run and recognise the “Term” and we got the results we expected.so our TPascal program was able to pass and translate the general expression that involves addidtion and subtraction***.***