Question 2

The post fix stack program features 2 functions called “OperandCheck”, and “EvaluateExpression”. Main begins with creating a string called “exp” and then later passes it on as a parameter to the EvaluteExpression function. Essentially all of the work is done inside of this function. A temporary stack is made to hold the digits apart of the string that is passed. A for loop is started from the end of the string to simulate the prefix stack. I use the the feature function OperandCheck which invokes the built-in C++ function “isdigit” and returns true if the current character is a digit. Going back to the EvaluteExpression function, if OperandCheck returns true, we will push it onto the stack because we only want numbers on the stack. If the current character is not a digit then the program will jump to the else statement in which the top two digits in the stack will be popped and the given expression will be executed on them. Once the given expression is done, the result will then be pushed onto the stack. At the end of the function we return the top of the stack because it will be the total result of the prefix expression once the program has completed its for loop. This question was fairly simple and straightforward. The hardest part was trying to understand the concept of a prefix stack vs a postfix stack because we are so use to a infix expression. Overall I would say that this was definitely on the easier side of the questions.