

Compile and Runtime Errors in Compiler

Jayashree. G¹, Suseela. S², Arisha. S³, Vinitha. K⁴

^{1,3}Third year students, ^{2,4}Assistant professors, Dept. of CSE

Periyar Maniammai Institute of Science and Technology, Vallam, Thanjavur,
Tamil Nadu, India.

ABSTRACT

Compiler and its error are the two fundamentals which bridges the gap between a programmer and the machine to work well on C. In this paper, we have shown about compiler and its error messages. We have also discussed about many recovery mechanisms in a compiler. As programming languages acts like an interface between a programmer and the machine, it should not be subjected to any error. If it supposed to possess error, then the code will not attain efficiency, meaning and quality. So some means of gap has to be bridged between the machine and to the user. This is where a compiler comes in. Here the task of a compiler is to compile the program or instruction which is written in a particular source language and convert it into a target language via various phases available in the compiler. Meanwhile, the tasks of error handling process are to detect each error, report it to the user, and possibly make some repair to allow processing to continue. Finally, the purpose of this paper is to provide an entire knowledge about the Compiler and its error briefly.

Keywords: Compilers; Errors; Target language.

I. INTRODUCTION

Mostly computer professionals won't write any compiler. Instead, a compiler translates (or compiles) a program written in a high-level programming language that is suitable for human programmers into the low-level machine language that is required by computers. So simply, compiler is a program that is designed to convert human readable higher-level programming language into machine language, or source code. When these programs are converted from one form to another the compiler may face some error. Compilation error refers to a stage where a compiler fails to perform compilation either due to errors in the code or, due to errors in the compiler itself. An error message often helps programmers to debug source code. Different types of errors are analysed and reported to the user. The main requirement for the compiler is to stop and report a message, and cease compilation. There are some common recovery methods:

1. Panic mode recovery: Basically, it prevents the parser from developing infinite loops while recovering error and this is the easiest way of error recovery. The parser discards the input symbol one at a time until one of the designated (like end, semicolon) set of synchronizing

tokens is found. This is enough when the presence of multiple errors in same For example: Consider the erroneous expression- (1 + + 7) + 4. Panic mode recovery method will skip ahead to next integer and then continues.

2. $E \rightarrow \text{int} | E + E | (E) | \text{error int} | (\text{error})$

3. Phrase level recovery: Error correction is a tedious process in this strategy. But, it performs local correction on the input to repair the error.

Example: Performs local correction by inserting a semicolon.

4. Error productions: There are some common errors known to the compiler designers that may occur in the code. Augmented grammars can also be used, as productions that generate erroneous constructs when these errors are encountered.

Example: write 2x instead of 2*x.

5. Global correction: The objective of global correction is to make as few changes as possible while converting an incorrect input string to a valid string. This strategy costs more to implement.

Example: When an erroneous input statement A is fed, it creates a parse tree for some closest error-free statement B.

statement is rare.

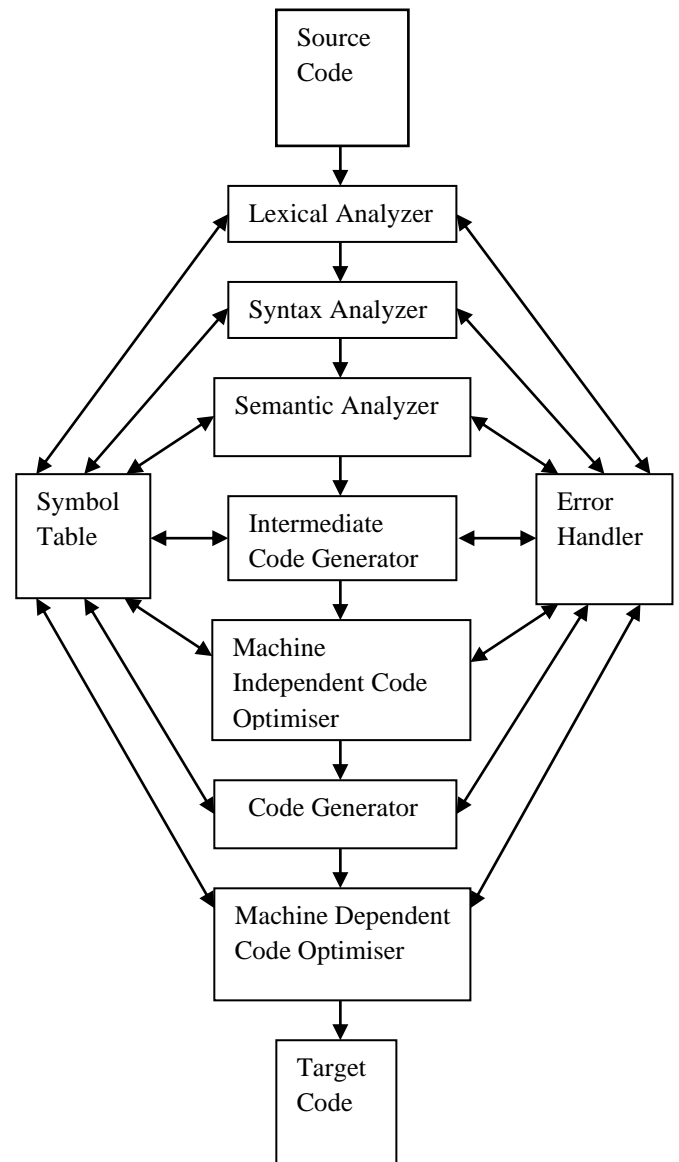


Figure 1. Phases of Compiler

Each phase in a compiler (Figure 1) may face error are shown below:

Lexical analyzer: Tokens are spelled wrongly.

Syntax analyzer: Missing parenthesis, comma etc.

Intermediate code generator: Mismatched operands for an operator.

Code Optimizer: When the statement is not reachable.

Code Generator: Unreachable statements.

Next in Section II. various types of errors are revised. Then in section III, analysis of error is discussed. In section IV, Many real time applications of compiler technology have been revised; summarization of this paper is being portrayed in section V. Finally the last section VI. of this paper includes references from which this paper is being written.

II. TYPES OF ERROR

This paper portraits a clear view about the architecture of types of error in a C Compiler via error handling mechanism. It is a process which is used to determine each error, report it to the user and then the process make some recover strategy and implement them to handle error. Major types of error: run-time and compile time error:

Due to invalid input data or adverse input parameters, the error which takes place during the execution of a program is run time error. So the executed code does not produce the desired result.

Example: The lack of sufficient memory to run an application or a memory conflict with another program and logical error.

An error that occurs during compile time is called compile time error and it happens before the execution of the program.

Example: Syntax error or missing file reference that prevents the program from successfully compiling.

Compile-time errors:

1. Lexical: It is a sequence of characters that does not match the pattern of any token

Example: Misspellings of identifiers, keywords or operators are included in this category.

2. Syntactical: Wrong syntax usage will cause this kind of error to occur. During execution this error will occur.

Example: Omitting the required semicolon, using an undeclared variable.

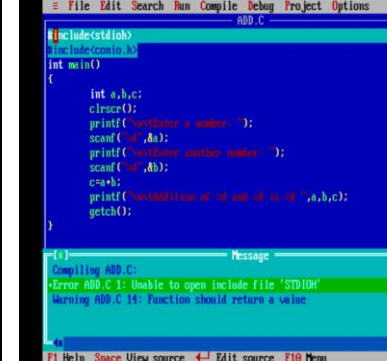
3. Semantical: If the meaning of any natural language is gets mismatched then this error may occur.

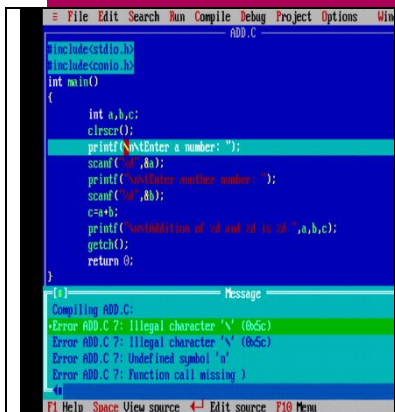
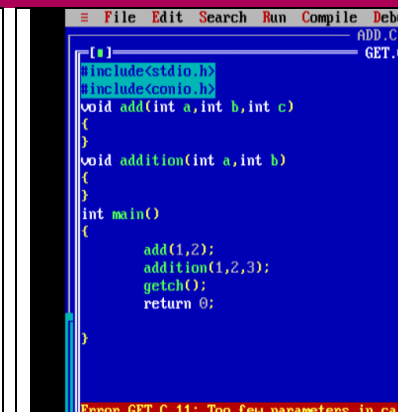
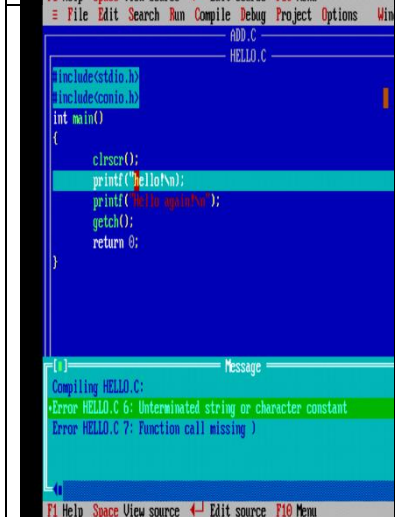
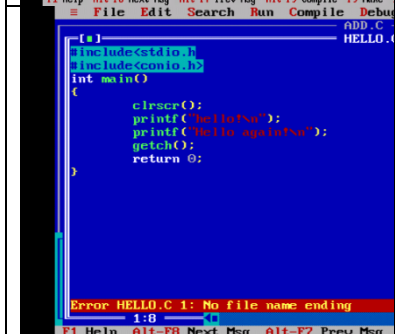
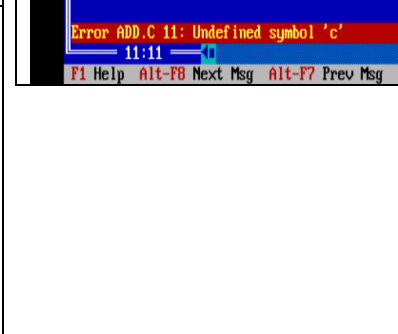
Example: Incompatible value assignment or type mismatches between operator and operand.

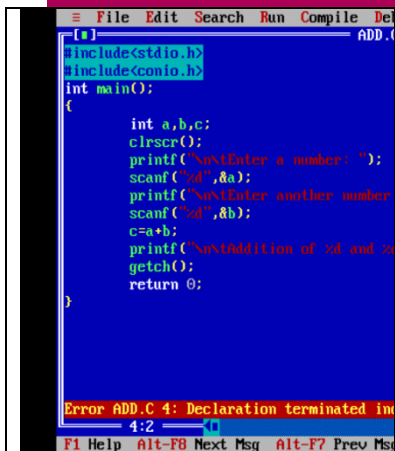
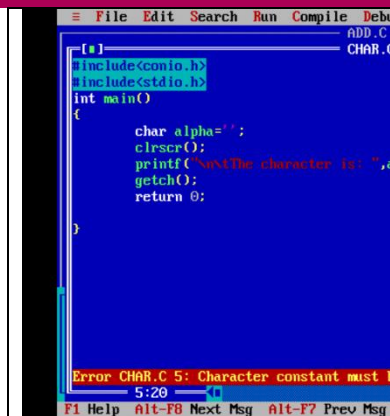
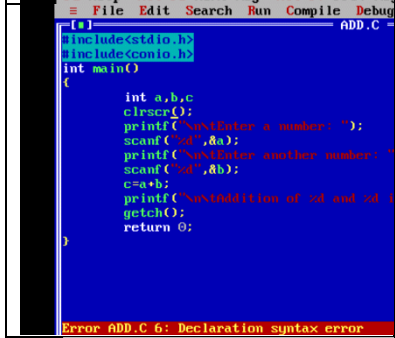
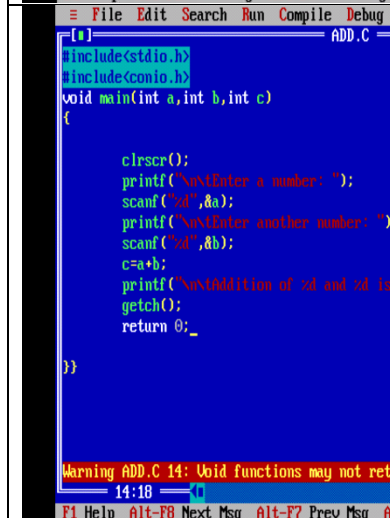
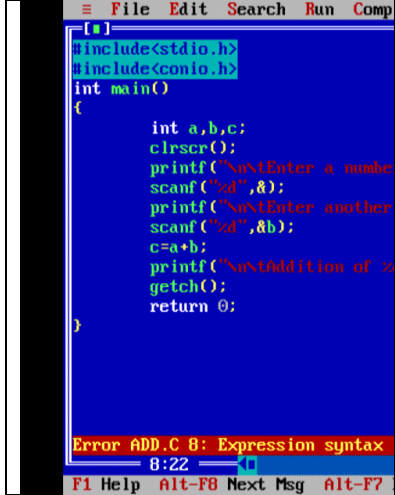
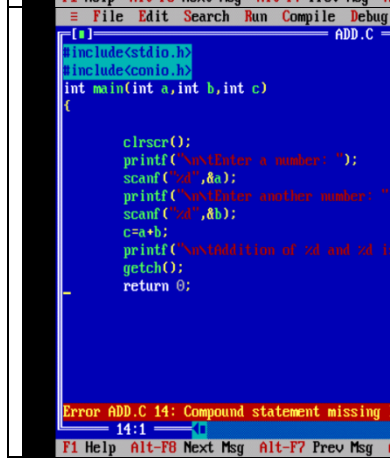
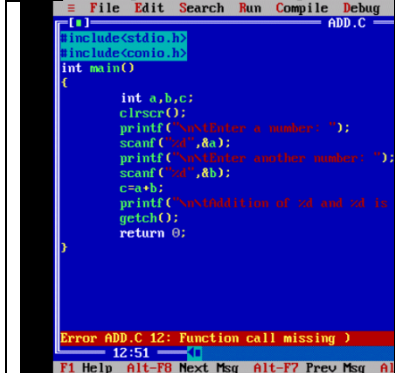
4. Logical: infinite loop, code not reachable.

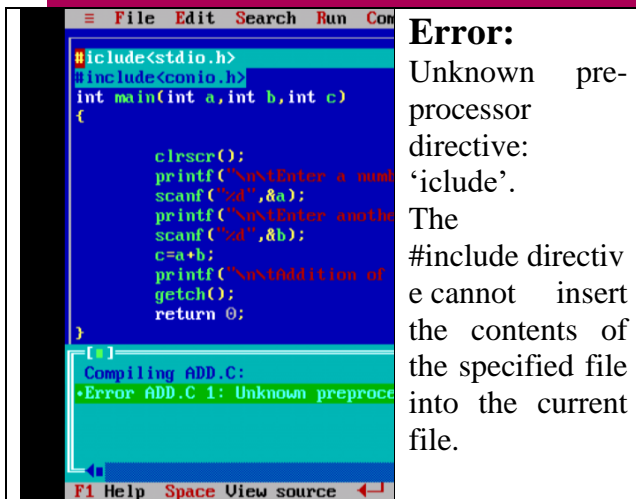
Example: Division by zero.

III. ERROR ANALYSIS

	<p>Error: Unable to open include file. This problem occurs due to wrong configuration in Turbo C++ directories.</p>
--	--

	<p>Error: Illegal character.</p> <p>This error occurs when there is an invalid or unexpected token that doesn't belong at this position in the code.</p>		<p>Error: Too few parameters in call to ' '.</p> <p>If very few parameters are created regardless of the declared variable then this error occurs.</p>
	<p>Error: Unterminated string or character constant.</p> <p>The compiler found no terminating quote after the beginning of a string or character constant.</p>		<p>Error: Declaration is not allowed here.</p> <p>This kind of error occur when the declaration is done after the call to clrscr() is made.</p>
	<p>Error: Statement missing.</p> <p>If any semicolon is missed at the end of the line then the error prevails.</p>		<p>Error: Undefined symbol.</p> <p>The link-editor searches the internal symbol table for any symbol references that have not been bound to symbol definitions.</p>
	<p>Error: No file name ending.</p> <p>The file name in a #include statement was missing the correct closing quote or angle bracket.</p>		<p>Error: Undefined symbol.</p> <p>The link-editor searches the internal symbol table for any symbol references that have not been bound to symbol definitions.</p>

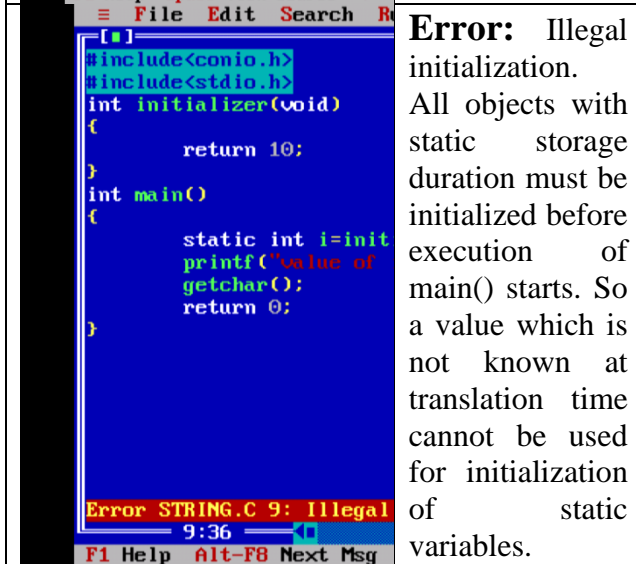
	<p>Error: Declaration terminated incorrectly.</p> <p>When the main function is called an additional semicolon paves the way for an error to occur.</p>		<p>Error: Character constant must be one or two characters long.</p> <p>Inside the apostrophes, none of the character is available.</p>
	<p>Error: Declaration syntax error.</p> <p>At the end of variable declaration, a semicolon is missed.</p>		<p>Error: Void functions may not return a value.</p> <p>When a function has arguments, it receives any data from the calling function but it returns no values.</p>
	<p>Error: Expression syntax.</p> <p>If an error occurs in a syntax containing tokens or characters, then it is syntax error.</p>		<p>Error: Compound statement missing.</p> <p>If the statements are not enclosed with the braces then the compound statement missing error occurs.</p>
	<p>Error: Function call missing.</p> <p>If the function is not been properly called then the function call is missed.</p>		



```
#include<stdio.h>
#include<conio.h>
int main(int a,int b,int c)
{
    clrscr();
    printf("NextEnter a num");
    scanf("%d",&a);
    printf("NextEnter another");
    scanf("%d",&b);
    c=a+b;
    printf("NextAddition of");
    getch();
    return 0;
}
```

Compiling ADD.C:
Error ADD.C 1: Unknown preprocessor directive: 'include'.

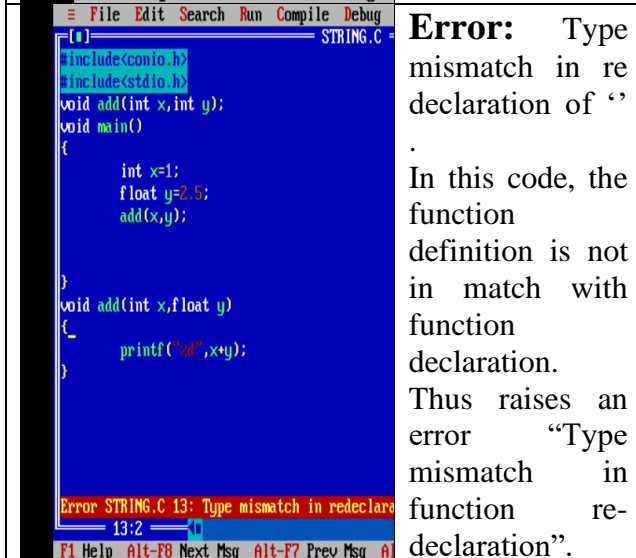
Error: Unknown pre-processor directive: 'include'. The #include directive cannot insert the contents of the specified file into the current file.



```
#include<conio.h>
#include<stdio.h>
int initializer(void)
{
    return 10;
}
int main()
{
    static int i=init
    printf("value of");
    getch();
    return 0;
}
```

Error STRING.C 9: Illegal initialization.

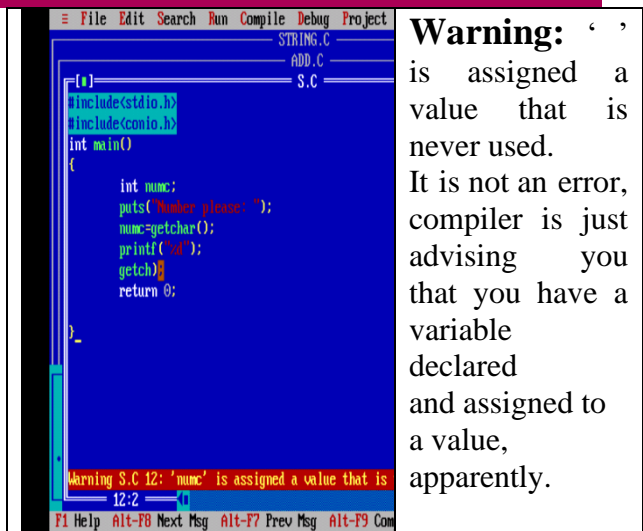
Error: Illegal initialization. All objects with static storage duration must be initialized before execution of main() starts. So a value which is not known at translation time cannot be used for initialization of static variables.



```
#include<conio.h>
#include<stdio.h>
void add(int x,int y);
void main()
{
    int x=1;
    float y=2.5;
    add(x,y);
}
void add(int x,float y)
{
    printf("%d",x+y);
}
```

Error STRING.C 13: Type mismatch in redeclaration

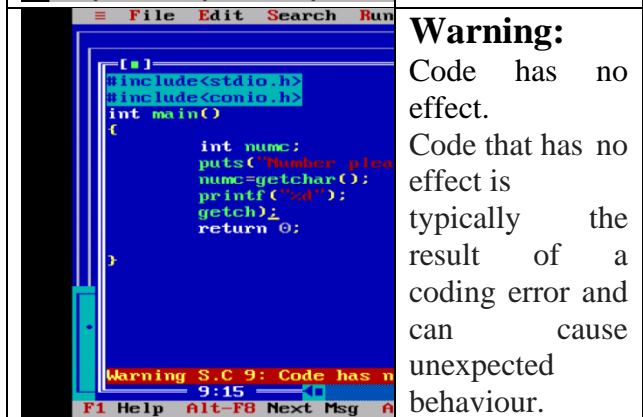
Error: Type mismatch in re declaration of ". In this code, the function definition is not in match with function declaration. Thus raises an error "Type mismatch in function re-declaration".



```
#include<stdio.h>
#include<conio.h>
int main()
{
    int numc;
    puts("Number please: ");
    numc=getchar();
    printf("%d");
    getch();
    return 0;
}
```

Warning S.C 12: 'numc' is assigned a value that is never used.

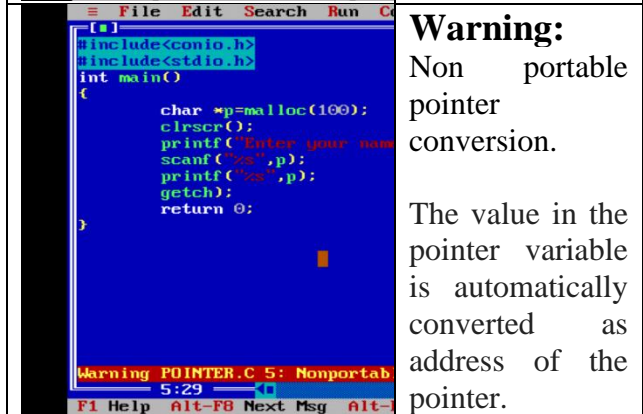
Warning: ' ' is assigned a value that is never used. It is not an error, compiler is just advising you that you have a variable declared and assigned to a value, apparently.



```
#include<stdio.h>
#include<conio.h>
int main()
{
    int numc;
    puts("Number please");
    numc=getchar();
    printf("%d");
    getch();
    return 0;
}
```

Warning S.C 9: Code has no effect.

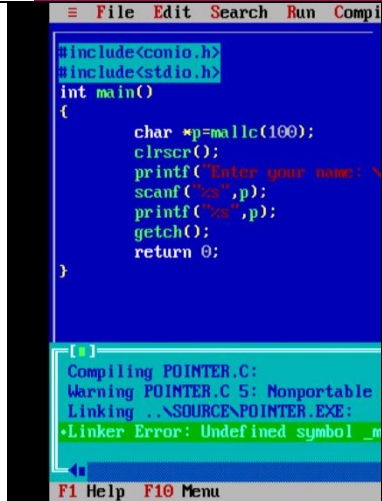
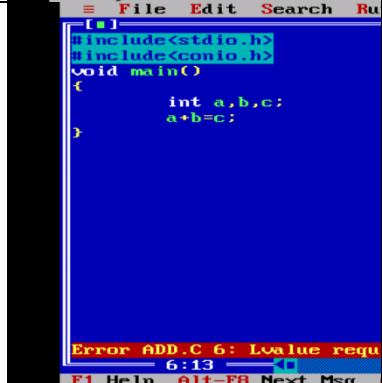

Warning: Code has no effect. Code that has no effect is typically the result of a coding error and can cause unexpected behaviour.



```
#include<conio.h>
#include<stdio.h>
int main()
{
    char *p=malloc(100);
    clrscr();
    printf("Enter your name");
    scanf("%s",p);
    printf("%s",p);
    getch();
    return 0;
}
```

Warning POINTER.C 5: Nonportable pointer conversion.

Warning: Non portable pointer conversion. The value in the pointer variable is automatically converted as address of the pointer.

	<p>Linker Error: Undefined symbol ' ' in module POINTER.C The code compiles fine, but that some function or library that is needed cannot be found.</p>
	<p>Error: Value required. This error occurs when the statements written in the program are not meaningful to the compiler.</p>
	<p>Warning: Possibly incorrect assignment. When the compiler encounters an assignment operator as the main operator of a conditional expression (part of an if, while, or do-while statement).</p>

- ✚ It delivers optimized computer architecture.
- ✚ This implements new software productivity tools.
- ✚ Lexical analyser techniques are used in ordinary text editors, pattern recognition programs etc.
- ✚ Most of the techniques used in compiler design are implemented in Natural Language Processing (NLP) systems.

V. CONCLUSION

Finally the purpose of this paper is to give a clear basic idea about the compiler, compiler error and error types along with its recovery mechanisms. In this, Compiler plays a major part which acts as an interface in converting one language into another language. It also includes classification of errors which makes the new programmers to understand the concept well. This again depicts the recovery mechanisms to prevent the occurrence of error. Also this compiler technology has other important uses as well. It works better for implementing high-level programming, Optimizing and designing of computer architectures, Software productivity tools like type and bounce checking. Meanwhile, widely it is known for program translations. Thus the main motive of this paper is to give a basic knowledge about the compiler and its error mechanisms to recover the error that has been occurred while coding.

VI. REFERENCES

[1]. H. C. A. V. Jatin Chhabra, "Research paper on Compiler Design," International Journal of Innovative

IV. APPLICATIONS OF COMPILER TECHNOLOGY.

- ✚ Compiler is used to convert a particular language into a target language. So this implements high level programming language.

Research in Technology, vol. 5, no. January 2019, pp. 1-2, 2014.

[2]. M. Rouse, "SearchSoftwareQuality," 2007. [Online]. Available: <https://searchsoftwarequality.techtarget.com/definition/errorhandling>. [Accessed 08 January 2019].

[3]. P. Agrawal, "geeksforgeeks," [Online]. Available: <https://www.geeksforgeeks.org/errorhandling-compiler-design/>. [Accessed 08 January 2019].

[4]. K. C. Louden, Compiler Construction: Principles and Practice, 1997.

[5]. Aho, Alfred V. and Johnson, Stephen C. [1976]. Optimal code generation for expression trees. Journal of the ACM, 23(3):488501.

[6]. Ross, D. T. [1967]. The AED free storage package. Communications of the ACM, 10(8):481492

[7]. Azevedo A, Issenin I, Cornea R, Gupta R, Dutt N, Veidenbaum A, Nicolau A dynamic voltage scheduling using program Design automation and test in Europe 2005.

[8]. <https://www.includehelp.com/c/too-few-arguments-to-function-c-language-error.aspx>

[9]. Aho, Alfred V., Hopcroft, J. E., and Ullman, Jeffrey D. [1974]. The Design and Analysis of Computer Algorithms. Addison Wesley, Reading, MA.

[10]. William M. Waite Department of Electrical Engineering University of Colorado Boulder, Colorado 80309 USA email: William.Waite@colorado.edu