Pruta Haldarmalani
44-58.
THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

EXPERIMENT NO. 9.

Aim: write a program to implement code Generation.

Theory:

code generation is the process of transforming an intermediate representation of a program into executable code in a target language or machine code

1] Intermediate Representation: Before code generation, the compiler typically transforms the source code into an intermediate representation (IR), which abstracts away from the specific source language syntax while retaining its semantic.

2] Target code generation: During code generation, the compiler traverses the IR & emits target code instruction or construct that perform equivalent operation to those described by the IR.

3] Target platform considerations: code generation takes into account the characteristics of the target platform, including its instruction set architecture (ISA), memory model, calling conventions & optimization opportunities.

4] Optimization: Optionally, code generation may involve optimization technique to improve the efficiency or quality of the generated code, such as constant folding, loop unrolling or register allocation.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Example:
simple expression to be compiled into machine code.

```
int add(int a, int b) {
    return a+b;
}
```

Intermediate Representation (IR):

```
Function add
PRAM a
PRAM b
TEMP t1 = a+b
RETURN t1
```

Target code generation (X86 assembly):

```
add:
    PUSH EBP        ; save the current base pointer
    MOV EBP, ESP    ; set up a new base pointer
    MOV EAX, [EBP+8]  ; load 'a' from stack into EAX
    ADD EAX, [EBP+12] ; Add 'b' to EAX
                      ; Result is in EAX.
    POP EBP         ; Restore the previous base pointer.
    RET             ; return from function, result is in EAX
```

18/3/24

A-1