1. Variable Declarations

Input: Simple variable declaration using standard data types.

int x = 10;

float y = 3.14;

char c = 'A';

```
Enter your code as a whole block (type 'END' on a new line to finish):
int x = 10;
float y = 3.14;
char c = 'A';
ENDint x = 10;

float y = 3.14;

char c = 'A';

END

Generated Intermediate Code (Three-Address Code):
x = 10;


=== Code Execution Successful ===
```

2. Simple Assignment Statements

Input: Variable assignments after declaration.

int x = 10;

x = x + 5;

```
Enter your code as a whole block (type 'END' on a new line to finish):
int x = 10;
x = x + 5;
ENDint x = 10;

x = x + 5;

END

Generated Intermediate Code (Three-Address Code):
x = 10;
x = x


=== Code Execution Successful ===
```

3. Conditional Statements

Input: Handling if conditions.

int x = 10;

if (x > 5) x = x + 1;

```
/tmp/FBRQ79KmGb.o
Enter your code as a whole block (type 'END' on a new line to finish
int x = 10;
if (x > 5) x = x + 1;
ENDint x = 10;

if (x > 5) x = x + 1;

END

Generated Intermediate Code (Three-Address Code):
x = 10;
if (x L1 > goto
goto
L1:


=== Code Execution Successful ===
```

4. Multiple Statements

Input: Multiple statements inside a conditional.

```
int a = 20;

if (a < 30) {

   a = a + 1;

}
```

```
Enter your code as a whole block (type 'END' on a new line to finish):
int a = 20;
if (a < 30) {
    a = a + 1;
}
ENDint a = 20;

if (a < 30) {

    a = a + 1;

}

END

Generated Intermediate Code (Three-Address Code):
a = 20;
if (a L1 < goto
goto
L1:
a = a


=== Code Execution Successful ===
```

5. Looping Statements (Simple Form)

Input: Handling while loops.

```
int i = 0;

while (i < 10) {

   i = i + 1;

}
```

```
Enter your code as a whole block (type 'END' on a new line to finish):
int i = 0;
while (i < 10) {
    i = i + 1;
}
ENDint i = 0;

while (i < 10) {

    i = i + 1;

}

END

Generated Intermediate Code (Three-Address Code):
i = 0;
i = i


=== Code Execution Successful ===
```

6. Invalid Declarations (Syntax Error)

Input: Missing identifier in a declaration.

int = 10;

```
Enter your code as a whole block (type 'END' on a new line to finish):
int = 10;
END

Generated Intermediate Code (Three-Address Code):
= = 0


=== Code Execution Successful ===
```

7. Invalid Assignment (Syntax Error)

Input: Missing = in an assignment.

int x;

x 10;

```
Enter your code as a whole block (type 'END' on a new line to finish):
int x;
x 10;
ENDint x;

x 10;

END

Generated Intermediate Code (Three-Address Code):
x; = 0


=== Code Execution Successful ===
```

8. Missing Semicolon (Syntax Error)

Input: Missing semicolon at the end of an assignment.

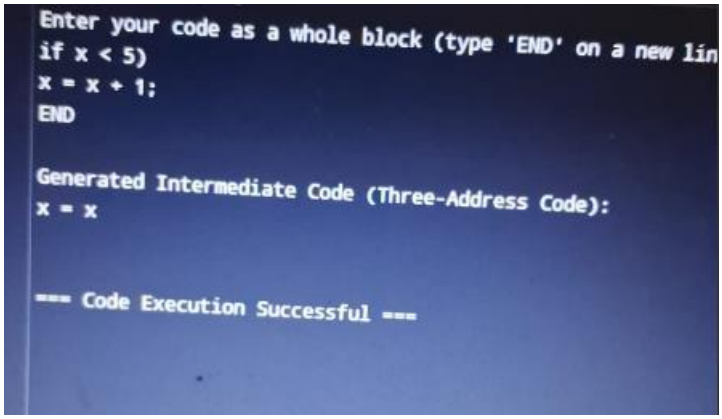
int x = 10

```
Enter your code as a whole block (type 'END' on a new line to finish):
int x = 10
END

Generated Intermediate Code (Three-Address Code):
x = 10


=== Code Execution Successful ===
```

9. Missing Parenthesis (Syntax Error in Condition)

Input: Incorrect parenthesis usage in a conditional.

if x > 5) x = x + 1;



10. Unclosed Block (Syntax Error in Loops/Conditionals)

Input: Unclosed braces for a block statement.

while (i < 10) {

i = i + 1;

```
Enter your code as a whole block (type 'END' on a new line to finish):
while (i < 10) {
    i = i + 1;
ENDwhile (i < 10) {

    i = i + 1;

END

Generated Intermediate Code (Three-Address Code):
i = i


=== Code Execution Successful ===
```

11. Simple Variable Declaration

int a;

float b;

```
int a;
float b;
ENDint a;

float b;

END

Generated Intermediate Code (Three-Address Code):
a; = 0


=== Code Execution Successful ===
```

12. Assignment Without Identifier

int = 10;

```
Enter your code as a whole block (type 'END' on a new line to finish):
int = 10;
END

Generated Intermediate Code (Three-Address Code):
= = 0


=== Code Execution Successful ===
```

14 Complex Expression

int sum = a + b * c / e - d;

```
Enter your code as a whole block (type 'END' on a new line to finish):
int sum = a + b * c / e - d;
END

Generated Intermediate Code (Three-Address Code):
sum = a


=== Code Execution Successful ===
```

15. Complex Expression

int a = b + c – d * e;

```
Enter your code as a whole block (type 'END' on a new line to finish):
int a = b + c - d * e;
END

Generated Intermediate Code (Three-Address Code):
a = b


=== Code Execution Successful ===
```

16. Invalid variable declaration

```
int main() {
   a % b;
}
```

```
/tmp/myCn2Fjxje.o
Enter your code as a whole block (type 'END' on a new line to finish):
int main() {
    a % b;
}
ENDint main() {

    a % b;

}

END

Generated Intermediate Code (Three-Address Code):
main() = 0


=== Code Execution Successful ===
```

17. Ignoring multiline comments

int sum = a + 4; /* This is a multi-line comment

that should be ignored */

```
/tmp/MpuNtvLpqj.o
Enter your code as a whole block (type 'END' on a new line to finish):
int sum = a + 4; /* This is a multi-line comment
                 that should be ignored */
ENDint sum = a + 4; /* This is a multi-line comment

                 that should be ignored */

END

Generated Intermediate Code (Three-Address Code):
sum = a


=== Code Execution Successful ===
```

18. Ignoring multiline comments

int sum = a + 4; // This is a comment

```
Enter your code as a whole block (type 'END' on a new line to finish):
int sum = a + 4; // This is a comment
END

Generated Intermediate Code (Three-Address Code):
sum = a


=== Code Execution Successful ===
```