SER 502: DESI COMPILER

Team Members:
Aneesh Dalvi (adalvi1)
Janice Abraham (jjabrah5)
Jinal Patel (jspatel5)
Viraj Talaty (vtalaty)

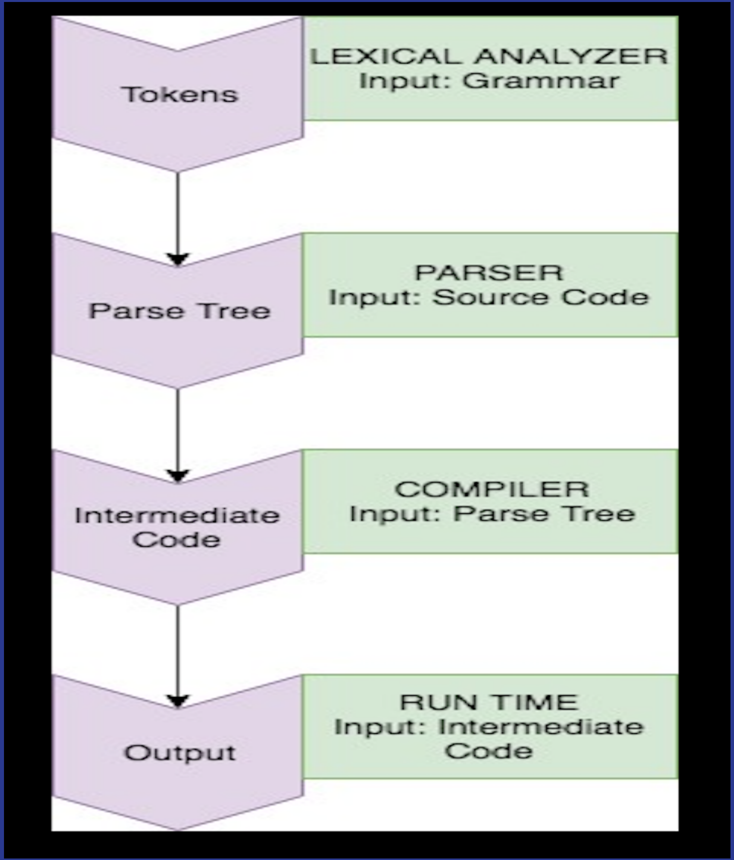
DESI supports...

Data Type	INT , BOOLEAN Type
Assignment	=
Decision Construct	If-Else-EndIf Loop
Iterative Execution	While Loop
Arithmetic Operators	+, -, *, /
Comparison Operator	==, !=, >, <, >=, <=
Logical Operator	&&, ||

Code Syntax Constants Explanation

shuru	{
khatam	}
int	Integer Value
boolean	Boolean Value - TRUE , FALSE
agar	if
magar	elseIf
nahitoh	else
bas	end-if
jabtak	while
tabtak	end-while
dikhao	print

Program Flow :



Grammar Rules:

```
grammar DesiGrammar;
program
            : 'shuru' block 'khatam'
            ;
block
            :
            | command+
            ;

command
            : (ifExpressn|whileExpressn|print|expression)
            ;
```

```
expression
            : 'int' IDENTIFIER (EQUALSto num_expressn)?
            | 'boolean' IDENTIFIER (EQUALSto bool_expressn)?
            | IDENTIFIER EQUALSto num_expressn
            | IDENTIFIER EQUALSto bool_expressn
    ;

bool_expressn
    : bool_expressn op=(ISEquals|NotEquals) bool_expressn
    | bool_expressn op=(AND|OR) bool_expressn
    | comp_expressn
    | '(' bool_expressn ')'
    | BOOLEAN

    ;
```

```
comp_expressn
    : num_expressn op=(GREATER|LESSER|MORE_or_EQU|LESS_or_EQU|ISEquals|NotEquals) num_expressn
    ;
num_expressn
    : num_expressn op=(MUL|DIV) num_expressn
    | num_expressn op=(ADD|SUB) num_expressn
    | '(' num_expressn ')'
    | SUB? DIGITS
    | SUB? IDENTIFIER
    ;
cond_expressn
    : '(' bool_expressn ')'
    ;
```

```
ifExpressn
    : 'agar' cond_expressn 'shuru' block 'khatam' (elseIfExpressn)*
(elseExpressn)? 'bas'
    ;

elseIfExpressn
    : 'magar' cond_expressn 'shuru' block 'khatam'
    ;

elseExpressn
    : 'nahitoh' 'shuru' block 'khatam'
    ;
```

```
whileExpressn
    : 'jabtak' cond_expressn 'shuru' block 'khatam' 'tabtak'
    ;

print
    : 'dikhao' '('
(DIGITS|BOOLEAN|IDENTIFIER|num_expressn|bool_expressn) ')'
    ;
```

```
DIGITS
        : [1-9] [0-9]*
        | '0'
        ;
BOOLEAN
        : 'True'
        | 'False'
        ;
```

```
ADD             : '+';
SUB             : '-';
MUL             : '*';
DIV             : '/';
AND             : '&&';
OR              : '||';
LESSER          : '<';
GREATER         : '>';
LESS_or_EQU     : '<=';
MORE_or_EQU     : '>=';
NotEquals       : '!=';
ISEquals        : '==';
```

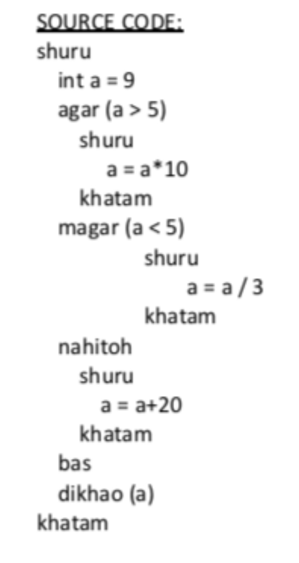
```
IDENTIFIER
            : [a-zA-Z_] [a-zA-Z_0-9]*
            ;

EQUALSto  : '=';

WS      : [ \t\r\n]+ -> skip; // skip spaces, tabs, newlines
Comment : ('//' ~[\r\n]* | '/*' .*? '*/') -> skip;
```

Intermediate Code Definition

```
IF_ELSE_SHURU = "IF_ELSE_SHURU";
IF_ELSE_KHATAM = "IF_ELSE_KHATAM";
IF_SHURU = "IF_SHURU";
IF_KHATAM = "IF_KHATAM";
ELSE_IF_SHURU = "ELSE_IF_SHURU";
ELSE_IF_KHATAM = "ELSE_IF_KHATAM";
ELSE_SHURU = "ELSE_SHURU";
ELSE_KHATAM = "ELSE_KHATAM";
DEFAULT_BOOL = "FALSE";
DEFAULT_INT = "0";
ACCUMULATOR_REGISTER = "ACC";
REGISTER_TWO = "B";
REGISTER_THREE = "C";
REGISTER_FOUR = "D";
STORE_INSTRUCTION = "LOAD";
WRITE_INSTRUCTION = "PRINT";
ADDITION = "JODNA";
SUBTRACTION = "GHATANA";
MULTIPLICATION = "GUNAKAR";
DIVISION = "VIBHAJNA";
UNARY_MINUS = "UNARY";
GT = "GT";
GTE = "GTE";
LT = "LT";
LTE= "LTE";
```

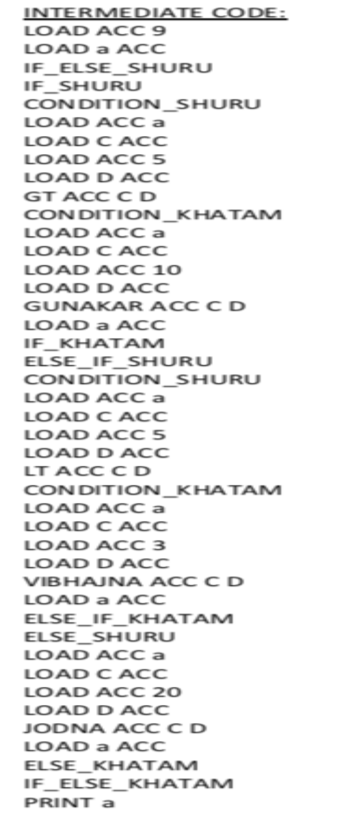
```
EQUAL_EQUAL = "EQUAL_TO";
NOT_EQUAL = "NOT_EQUAL_TO";
OR = "OR";
AND = "AND";
CONDITION_SHURU = "CONDITION_SHURU";
CONDITION_KHATAM = "CONDITION_KHATAM";
WHILE_SHURU = "WHILE_SHURU";
WHILE_KHATAM = "WHILE_KHATAM";
```

Sample Run 1



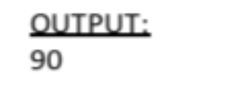
SOURCE CODE:

```
shuru
   int a = 9
   agar (a > 5)
      shuru
          a = a*10
      khatam
   magar (a < 5)
               shuru
                     a = a / 3
               khatam
   nahitoh
      shuru
         a = a+20
      khatam
   bas
   dikhao (a)
khatam
```



INTERMEDIATE CODE:

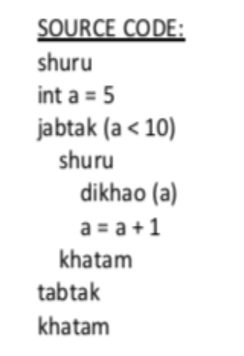
```
LOAD ACC 9
LOAD a ACC
IF_ELSE_SHURU
IF_SHURU
CONDITION_SHURU
LOAD ACC a
LOAD C ACC
LOAD ACC 5
LOAD D ACC
GT ACC C D
CONDITION_KHATAM
LOAD ACC a
LOAD C ACC
LOAD ACC 10
LOAD D ACC
GUNAKAR ACC C D
LOAD a ACC
IF_KHATAM
ELSE_IF_SHURU
CONDITION_SHURU
LOAD ACC a
LOAD C ACC
LOAD ACC 5
LOAD D ACC
LT ACC C D
CONDITION_KHATAM
LOAD ACC a
LOAD C ACC
LOAD ACC 3
LOAD D ACC
VIBHAJNA ACC C D
LOAD a ACC
ELSE_IF_KHATAM
ELSE_SHURU
LOAD ACC a
LOAD C ACC
LOAD ACC 20
LOAD D ACC
JODNA ACC C D
LOAD a ACC
ELSE_KHATAM
IF_ELSE_KHATAM
PRINT a
```



OUTPUT:

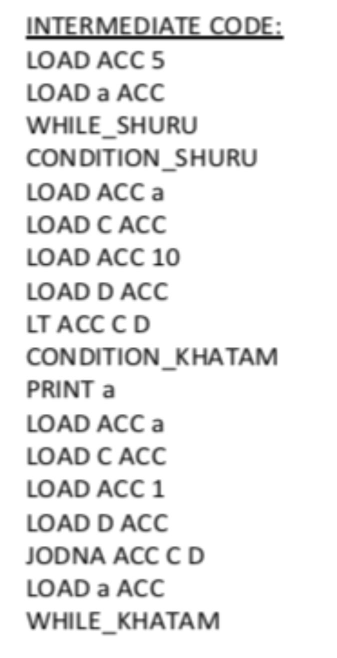
```
90
```

Sample Run 2



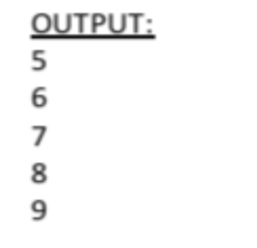
SOURCE CODE:

```
shuru
int a = 5
jabtak (a < 10)
   shuru
      dikhao (a)
      a = a + 1
   khatam
tabtak
khatam
```



INTERMEDIATE CODE:

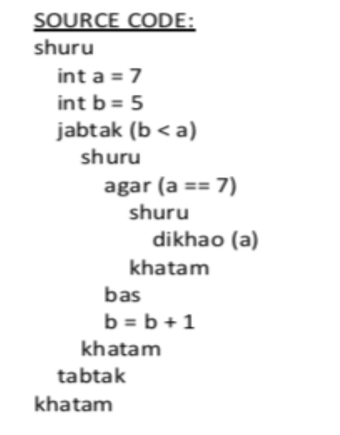
```
LOAD ACC 5
LOAD a ACC
WHILE_SHURU
CONDITION_SHURU
LOAD ACC a
LOAD C ACC
LOAD ACC 10
LOAD D ACC
LT ACC C D
CONDITION_KHATAM
PRINT a
LOAD ACC a
LOAD C ACC
LOAD ACC 1
LOAD D ACC
JODNA ACC C D
LOAD a ACC
WHILE_KHATAM
```



OUTPUT:

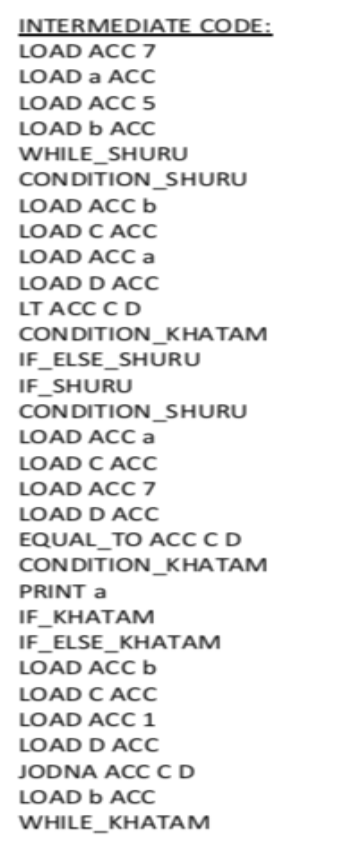
```
5
6
7
8
9
```

Sample Run 3



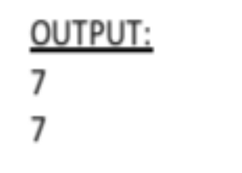
SOURCE CODE:

```
shuru
   int a = 7
   int b = 5
   jabtak (b < a)
      shuru
         agar (a == 7)
            shuru
               dikhao (a)
            khatam
         bas
         b = b + 1
      khatam
   tabtak
khatam
```



INTERMEDIATE CODE:

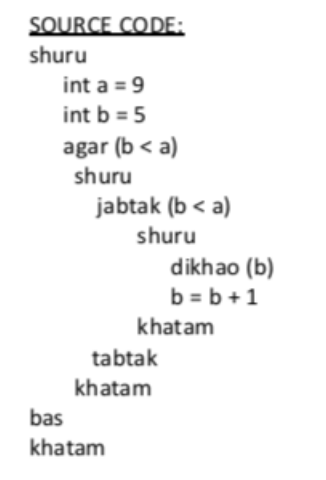
```
LOAD ACC 7
LOAD a ACC
LOAD ACC 5
LOAD b ACC
WHILE_SHURU
CONDITION_SHURU
LOAD ACC b
LOAD C ACC
LOAD ACC a
LOAD D ACC
LT ACC C D
CONDITION_KHATAM
IF_ELSE_SHURU
IF_SHURU
CONDITION_SHURU
LOAD ACC a
LOAD C ACC
LOAD ACC 7
LOAD D ACC
EQUAL_TO ACC C D
CONDITION_KHATAM
PRINT a
IF_KHATAM
IF_ELSE_KHATAM
LOAD ACC b
LOAD C ACC
LOAD ACC 1
LOAD D ACC
JODNA ACC C D
LOAD b ACC
WHILE_KHATAM
```



OUTPUT:

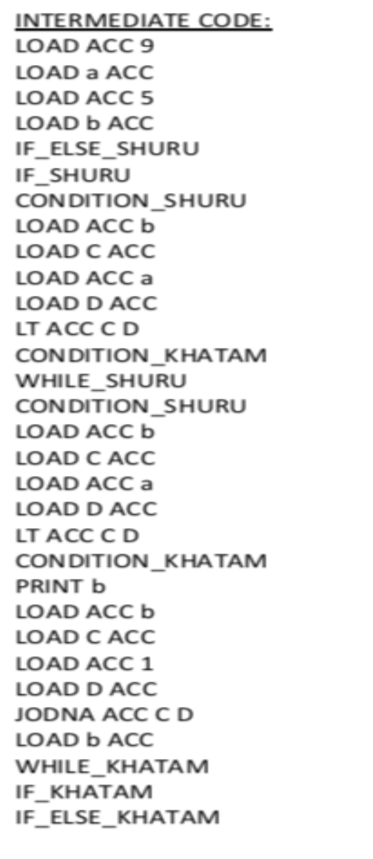
```
7
7
```

Sample Run 4



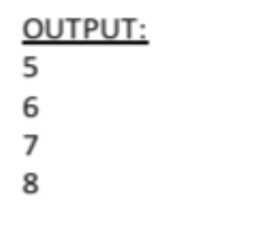
SOURCE CODE:

```
shuru
    int a = 9
    int b = 5
    agar (b < a)
     shuru
        jabtak (b < a)
             shuru
                 dikhao (b)
                 b = b + 1
             khatam
        tabtak
     khatam
bas
khatam
```



INTERMEDIATE CODE:

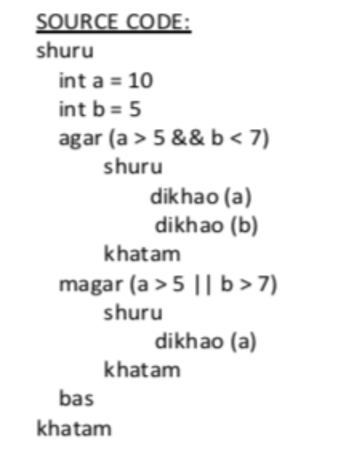
```
LOAD ACC 9
LOAD a ACC
LOAD ACC 5
LOAD b ACC
IF_ELSE_SHURU
IF_SHURU
CONDITION_SHURU
LOAD ACC b
LOAD C ACC
LOAD ACC a
LOAD D ACC
LT ACC C D
CONDITION_KHATAM
WHILE_SHURU
CONDITION_SHURU
LOAD ACC b
LOAD C ACC
LOAD ACC a
LOAD D ACC
LT ACC C D
CONDITION_KHATAM
PRINT b
LOAD ACC b
LOAD C ACC
LOAD ACC 1
LOAD D ACC
JODNA ACC C D
LOAD b ACC
WHILE_KHATAM
IF_KHATAM
IF_ELSE_KHATAM
```



OUTPUT:

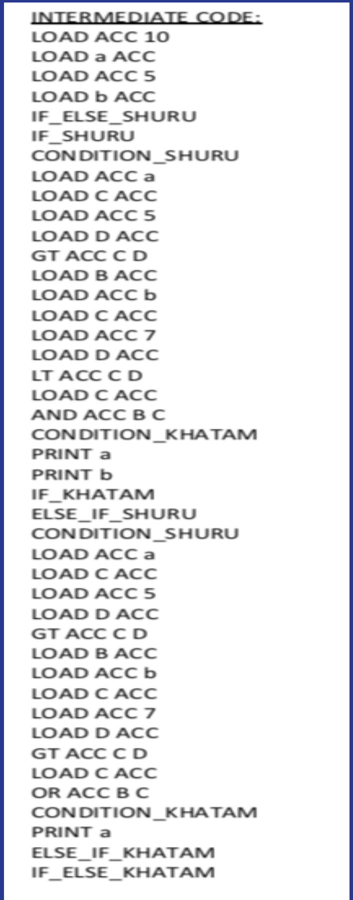
```
5
6
7
8
```

Sample Run 5



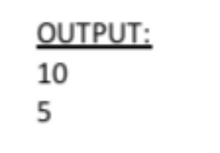
SOURCE CODE:

```
shuru
   int a = 10
   int b = 5
   agar (a > 5 && b < 7)
         shuru
                dikhao (a)
                 dikhao (b)
         khatam
   magar (a > 5 || b > 7)
         shuru
                 dikhao (a)
         khatam
   bas
khatam
```



INTERMEDIATE CODE:

```
LOAD ACC 10
LOAD a ACC
LOAD ACC 5
LOAD b ACC
IF_ELSE_SHURU
IF_SHURU
CONDITION_SHURU
LOAD ACC a
LOAD C ACC
LOAD ACC 5
LOAD D ACC
GT ACC C D
LOAD B ACC
LOAD ACC b
LOAD C ACC
LOAD ACC 7
LOAD D ACC
LT ACC C D
LOAD C ACC
AND ACC B C
CONDITION_KHATAM
PRINT a
PRINT b
IF_KHATAM
ELSE_IF_SHURU
CONDITION_SHURU
LOAD ACC a
LOAD C ACC
LOAD ACC 5
LOAD D ACC
GT ACC C D
LOAD B ACC
LOAD ACC b
LOAD C ACC
LOAD ACC 7
LOAD D ACC
GT ACC C D
LOAD C ACC
OR ACC B C
CONDITION_KHATAM
PRINT a
ELSE_IF_KHATAM
IF_ELSE_KHATAM
```



OUTPUT:

```
10
5
```

Future Work

- Adding new data types like Characters, Strings, Double
- Supporting data structures like Arrays, Lists, Maps
- Handling project structures with interfaces, overloading, polymorphism