LANGUAGE MANUAL

Contributors:

cs18b036 Sripranav Mannepalli

cs18b032 SRVS Maheswara Reddy

cs18b001 AVS Hrudai

cs18b006 Chirag Gupta

TABLE OF CONTENTS: **PGNO:**

PROGRAM CONSTRUCTS:

❖ Identifiers:

1. Variable Identifiers:

- Any combination of alphabets is allowed
  - *Example :* pqr , abcd
- Can contain 1 digit at max that too at the end
  - *Example :* pqr1 , abcd4

2. Function Identifiers:

- Function identifiers should begin with '_' (underscore) and any combination of alphabets is allowed.
  - *Example :* _pqr() , _abcd()
- Can contain 1 digit at max that too at the end.
  - *Example :* _pqr1() , _abcd4()

❖ Data Types :

- Integer : int
  - *Example : int a;*
- Floats : float
  - *Example : float b;*
- Characters :character
  - *Example : character c;*

❖ Arrays :

- 1Dimension Arrays (list):
  - We have 1 dimension arrays of integers called lists.
    - *Example :*

```
list a[5];        (declaring list of size 5)
a=[1;2;3;4;5;];          (assigning list)
```

- We can access i'th element as a[i].
  - *Example* :
    From the previous example,
    a[2] (3rd element of list , that is '3')

- 2Dimension Arrays (matrix)
  - We have 1 dimension arrays of integers called lists.
    - *Example :*

      list a[2][3];(declaring matrix of size 2x2)

      a=[1,2,3;11,12,13;] (assigning matrix)

❖ Operators:

- Operations on integers:
  - Arithmetic operations : + , - , * , /
  - Bitwise Operations : | , & , ^ , !

- Operations on floats:
  - Arithmetic operations : + , - , * , /

- Operations on lists(1D array):
  - + : adds two lists element wise
  - - : subtracts two lists element wise
  - @ : Returns size of list

- Example :

```
list a[3];
list b[3];
list c[3];
int s;
a=[1;2;3;]
b=[4;5;6;]
c=a+b ( this makes c as [5;7;9;] )
c=a-b ( this makes c as [-3;-3;-3;] )
```

```
s=@ a   ( this makes s as 3 because the size of
         list 'a' is 3 )
```

- Operations on matrix (2D array):
  - +: adds two matrices element wise
  - - : subtracts two matrices element wise
  - @ : Returns x dimension of matrix
  - @@: Returns y dimension of matrix
  - Example :

```
matrix  a[2][3];
matrix  b[2][3];
matrix  c[2][3];
int s;
a=[1,2,3;11;12,13;];
b=[4,5,6;14;15,16;];
c=a+b ( this makes c as [5,7,9;25;27,29;] )
c=a-b  ( this makes c as [-3,-3,-3;-3,-3,-3;] )
s=@ a ( this makes s as 3 because the x
dimension of a is 2 )
s=@@ a ( this makes s as 3 because the y
dimension of a is 3 )
```

❖ Expressions:

- Arithmetic :
  - `c = a + 5;`
  - `c = a + b;`
  - `c = a *3;`
  - `c = a/2;`
  - `c = a[0] + 2;`
  - `c[0] = a[0]*b[1];`

- Relational:
  - `a < 5;`
  - `a > b;`
  - `a <= b;`
  - `a[0] <= b[0];`
  - `a[0] < b;`
  - `a[0] >= 2;`

- Logical :
  - a || b;
  - a && b;
  - !a;
- Bitwise:
  - a|b;
  - a&b;
  - a^b;

❖ Iterative Loops:

- We support For and While loops are accepted.
- Nested iterative loops are also supported.
- **for loop:**
  - for(initialization;condition;iteration){
    CODE;
    }
  - Variable used in for looping has to be declared before For Loop.
    - Example:

```
int i = 0;
for(i = 0; i < 10; i++){
    print(i);
}
```

- **while loop:**
  - while(condition){
    CODE;
    }
  - Example:

```
int i ;
i=10;
while(i>0){
    print(i);
```

```
        i--;
    }
```

❖ Conditional statements:

- We support if and if-else conditional statements.
- Nested COnditional statements are also supported.
- **if :**
  - if(condition){
    CODE;

  - Example :
    ```
    if(a<b && b>c ){
        a=b+c;
    }
    ```
- **if-else:**
  - if(condition){
    CODE;
    }
    else
    CODE;
    }
  - Example :
    ```
    if(a<b ||  b>c ){
        a=b+c;
    }
    else{
        a=b-c;
    }
    ```

❖ Comments:

- Our language supports comments in the program.
- Comments must start with "#" and should be single lined.
- Examples:
  - `int a;` # We are defining an integer

❖ **Input/Output:**

- Print statement:
  - We can print the values of identifiers:
  - We can print integer, float and character values.
  - Example:

```
int a;
a = 5;
print(a);
```

- Scan statement:
  - We can scan integer, float and character values.
  - Example:

```
int a;
scan(a);
```

❖ **Functions:**

- FunctionType Name(parameters).
- Function can take any number of arguments.
- Function should return its datatype only.
- Our language also supports recursive functions.
- Examples:

```
int _fibonacci(int a){
    if(a<=1){
        return a;
    }
    else{
        int b;
        int c;
        b=fibonacci(a-1);
        c=fibonacci(a-2);
        return b+c;
    }
}
```

SAMPLE CODES :

- A Program which checks whether a list is palindrome or not.

```
# Check if a list is Palindrome

int _main(){
    int siz;
    list palin[9];
    palin=[1;2;3;4;5;4;3;2;1;];
    int ispalin;
    ispalin=1;
    int i;
    int t;
    int m;
    int n;
    siz=@palin;
    for(i=0;i<siz;i++;){
        t=siz;
        t=t- 1;
        t=t- i;
        m=palin[i];
        n=palin[t];
        if(m!=n){
            ispalin=0;
        }
    }
    print(palin);
    print(ispalin);
}
```

- A programme that sorts the list using bubble sort.

```
test_BubbleSort1

Int _main(){
    int siz;
    List pran[7];
    pran=[19;35;4;78;46;23;5;];
    siz@pran;
    int i;
    int j;
    int t;
    int m;
    int n;
    int o;
    siz--;
    for(i=0;i<siz;i++;){
        t=siz-i;
        for(j=0;j<t;j++;){
            m=pran[j];
            o=j+1;
            n=pran[o];
            if(m>n){
                m=pran[j];
                n=pran[o];
                pran[j]=n;
                pran[o]=m;
            }
        }
    }

    print(pran);
}
```

- A Program which takes a list (1D Array) Input from the user and prints the sorted list using bubble sorting.

```
# Lists - Sorting ( 1D Arrays of
Integers)

int _main(){
    int size;
    int m;
    int n;
    int o;
    list mylis[10];
    siz=@mylis;
    int i;int j;int t;

    for(i=0;i<size;i++;){
        scan(j);
        mylis[i]=j;
    }

    print(mylis);

    siz--;

    for(i=0;i<size;i++;){
        t=size-i;
        for(j=0;j<t;j++;){
            m=mylis[j];
            o=j+1;
            n=mylis[o];
            if(m>n){
                m=mylis[j];
                n=mylis[o];
                mylis[j]=n;
                mylis[o]=m;
            }
        }
    }
    print(mylis);
}
```

- A Program which calculates power of a number using function-recursion

```
int _main(){
    int aa;
    aa=2;
    int bb;
    bb=10;
    int k;
    k=_power(aa,bb);
    print(k);
}
int _power(int a,int b){
    int cc;
    cc=a;
    int t;
    if(b==1){
        return cc;
    }
    else{
        b=b- 1;
        t=_power(a,b);
        cc=cc*t;
        return cc;
    }
}
```

- A programme that prints fibonacci series given the number of elements to be printed.

```
int _main ()
{
    int n;
    scan(n);
    int ans;
    ans=_fibonacci(n);
    print(ans);
}

int _fibonacci(int n)
{
    int a;
    int b;
    int c;
    int i;
    a=0;
    b=1;
    if(n==0){
        b=0;
    }
    for(i=2;i<=n;i++;){
        c=a+b;
        a=b;
        b=c;
    }
    return b;
}
```

- A programme that finds GCD of two integers.

```
# GCD of two integers

int _main(){
    int n;
    scan(n);
    int m;
    scan(m);
    while(m!=n){
        if(m>n){
            m=m-n;
        }
        else{
            n=n-m;
        }
    }
    print(m);
}
```

- A program involving float data type.

```
int _main(){
    float a;
    float b;
    float c;

    a=123.45;
    b=456.78;
    c=a+b;
    print(c);
    c=a-b;
    print(c);
    c=a*b;
    print(c);
    c=a/b;
    print(c);
}
```

- A program to show all the conditional and iterative statements with nesting.

```
int _main(){
    int a;
    int b;
    b=5;
    for(a=0;a<=20;a++){
        if(a<10 && a>5){
            while(b>0){
                print(b);
                b--;
            }
        }
        else{
            if(a>10){
                b=10;
            }
            else{
                b=5;
            }
        }
    }
}
```