# Robotic Simulation Language - Specifications

Primitive Datatypes:

int , float , boolean

Complex(Custom) Datatypes:
Point, Velocity (internal structure same as Point but different units) , Bot

```
struct Point
{
        int x;
        int y;
};

struct Bot
{
        Point position;
        Velocity vel;

        int direction;
        int color;
        boolean active;
}
;
```

Aggregative Datatypes:
- Array of char, int, float
- 2-D arrays supported

- User defined structures supported
- Nested structures supported

Operators:

| | | |
|---|---|---|
| arithmetic | ==> | + - / * |
| unary prefix | ==> | ++ -- |
| shorthand | ==> | += |
| logical | ==> | AND(&&) OR( \| \| ) NOT(!) |
| relational | ==> | > < >= <= != == |

| | | | |
|---|---|---|---|
| array splicing | ==> | **:** | // Eg. b[2:4] = arr[3:5] |
| addV | ==> | addV v1, v2 | // for adding two velocities |
| input operator | ==> | >> | // for user input |
| forward operator | ==> | fw b1, 5 | // to move the bot forward |
| turn operator | ==> | rt b1 | // to turn the bot right |
| special assignment | ==> | := | // used for Point, Bot and velocity assignment |

<u>Other Features:</u>
static typing

static scoping

weak name equivalence for primitive data types, strong name equivalence for user defined structures

Type conversion, casting : Both implicit and explicit

Precedence in expressions taken care of

<u>Statements:</u>

<u>1. Iteration ==> For Loops</u>

- *for(i=5;i<6;++i;) {}*
- **nested** loops supported

<u>2. Functions</u>

*function <return_type1, return_type2, ....>*
*function_name (params) {*
*// body*

*return val1, val2, val3..;*

*}*

**- Multiple return values supported**

**- Parameters: Name based and Positional matching, Default values on right side**

3. Conditional

```
if (condition) {

}

else if (condition) {

}

else{

}
```

5. Assignment

Multiple parallel assignments support

eg. a, d, e = 4, 7, getVal();

4. Declaration

<type> <var_name> = <value>

5. return, break, continue, exit()
- break, continue only allowed inside loops

6. User Input
- readi >> x;
- support for user to input integers