The purpose of this manual is to guide a user through the usage of JTOB (Just Trying Our Best). JTOB (pronounced Jay-tawb) isn't necessarily a robust language, so this manual should be quite short. The language was based off the widely known and implemented Looplang language. It was created by Ryan Custard, Tim Schlottman, Steven Wall, Sean Poston, Logan Emel, and Qizheng Ma. It wasn't created with utmost simplicity in mind, but rather as a challenge to any programmer who wishes to do more with less. After all, less is more.

The most basic and important part of a language is variable manipulation. This is done in JTOB with:

```
am arrayName[1] = 5
```

The `am` keyword here is paramount to the assignment as it lets the compiler know that it's not checking Boolean conditions.

Another key feature of JTOB is the usage of arrays. To declare one:

```
array [5] arrayName
```

This will declare an array with size of 5 named arrayName.

Now on to printing:

```
say array[3]
```

The arithmetic in JTOB is straightforward:

```
5 + 3
```

```
5 - 3
```

```
5 * 3
```

```
5 / 3
```

Boolean operators are also included:

```
5 < 3
```

```
5 > 3
```

```
5 <= 3
```

```
5 >= 3
```

These operators can be used in conditional statements to run if statements:

```
if 5 < 3:
      am variableName = 8
else if 4 >= 3:
      am variableName = 9
else:
```

```
	am variableName = 10
```

JTOB also includes while loops:

```
while 5 > 3:
	am arrayName[3] = 5
end
```

Of course, any language worth its salt will include for loops:

```
floop 1 to 5:
	say arrayName[1]
end
```

To gain user input:

```
insert arrayName[1]
```

The full extent of the language can be recognized through the use of these commands. While JTOB does lack many of the capabilities that other languages offer, it's still able to perform many tasks, and its simplicity is what gives it its rugged charm.