Student Program Flow Control:

```python
class Student:
    def __init__(self, name, year):
        self.name = name
        self.year = year

    def getName(self):
        return self.name

    def getYear(self):
        return self.year

    def setYear(self, newyear):
        self.year = newyear

newstudent = Student("Lane", 2021)
newstudent.setYear(2022)
```

Memory Analysis:

- Before executing new student assignment line:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Value</td>
</tr>
<tr>
<td>Self</td>
<td>Value</td>
</tr>
<tr>
<td>Self</td>
<td>Value</td>
</tr>
</tbody>
</table>

- Execute du = new Student code new Student object using its class definition.

- Like a factory product just another item of its own reference design.

- `init_` is the constructor.

- `Self` becomes a reference to the Student object.

- Execute lines in constructor.

  - Recall that dot notation allows us to travel or teleport between symbol tables.
  
  - `Self` name, `Self` year assign changes value in Student object.

- Back to main code.

  - `init_` symbol table destroyed.

- But Student object line assigned to new Student:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Value</td>
</tr>
<tr>
<td>Self</td>
<td>Value</td>
</tr>
</tbody>
</table>
  | Self | Year | 2021

- `print(newStudent.getYear())`