**Valid and Invalid Variable Names:**

Within python, not all variable names used will be valid for use in running a program. Legal variable names accept capitalisation and underscores as perfectly valid properties by which to define them. Variable names can also have numbers in them, but the name must always start with a letter. The biggest issue to watch out for in naming is the use of *Keywords*. These are words that already have a function or purpose pre-allocated to them. The “print” function is an example of this, as “Print” can be used as a variable, but “print” cannot. There is a list below, following these concepts:

|  |  |  |
| --- | --- | --- |
| **Valid** | **Invalid** | **Reason** |
| Print | print | “Print” is not a predefined function. |
| Test\_01 | 01\_Test | Variable cannot start with numbers. Underscores also valid. |
| Test\_First | \_Test | Variable must start with letters. |
| prints | print | “prints” is not a predefined function and non-capital letters are permitted. |
| A1 | 1A | Variable cannot start with numbers. |

**Good and Bad Variable Names:**

|  |  |  |
| --- | --- | --- |
| **Good** | **Bad** | **Reason** |
| Variable\_01 | Variable01 | Use of the underscore makes the variable look neater and easier to read what’s what. Underscore easily shows the numbers as a defining trait. |
| Test | T | Using incredibly simple names that could correlate to anything is highly confusing when checking code. Should be labelled better so it can be understood when needed. |
| Category\_Item\_0354 | RndmItem | Use of clearly separating the variable into different parts that can be used reference it much easier among many similar variables. The other is a random sequence of text, which can be hard to find amongst lots of code. |