IS280: Python Programming

Final: Depreciation Calculator 2021

For this assignment you will develop a depreciation calculator using an external class that can return values for annual depreciation cost and values to build a schedule for the life of the asset.

Depreciation is an accounting concept that assigns yearly cost allocations for assets such as equipment and other material resources. That is, if a company purchases a piece of equipment for $30,000 but expects to use that piece of equipment for several years, it cannot take the full expense of the item in the year it is purchased – the expense must be portioned out over the life of the item.

The simplest form of depreciation is called straight line (SL). In this form the same amount of expense is recorded each year. Depreciation expenses are always based on the original purchase cost of an item less any 'salvage value' that is expected at the end of the useful life of the item. In straight line depreciation, the difference between cost and salvage value is just divided by the number of years of service expected. In other words, if our $30,000 truck is expected to be worth $5000 after 5 years of useful life, then the depreciation per year is (30000 - 5000) / 5 = $5000 per year. A straight line depreciation schedule would be:

Year	Start Value	Depreciation	End Value
1	30,000	5,000	25,000
2	25,000	5,000	20,000
3	20,000	5,000	15,000
4	15,000	5,000	10,000
5	10,000	5,000	5,000

However, a more advantageous type of depreciation is called Sum of the Year's Digits (SYD). It allows more of the cost of the item to be taken in early years (i.e., 'Today's Dollars'). In this method, you add up the numerical values of the number of years of the asset's life, and then use the 'life remaining' divided by that sum to produce the depreciation rate in a given year. With this approach, you apply the calculated rate to the total expected depreciation amount (i.e., cost – salvage value).

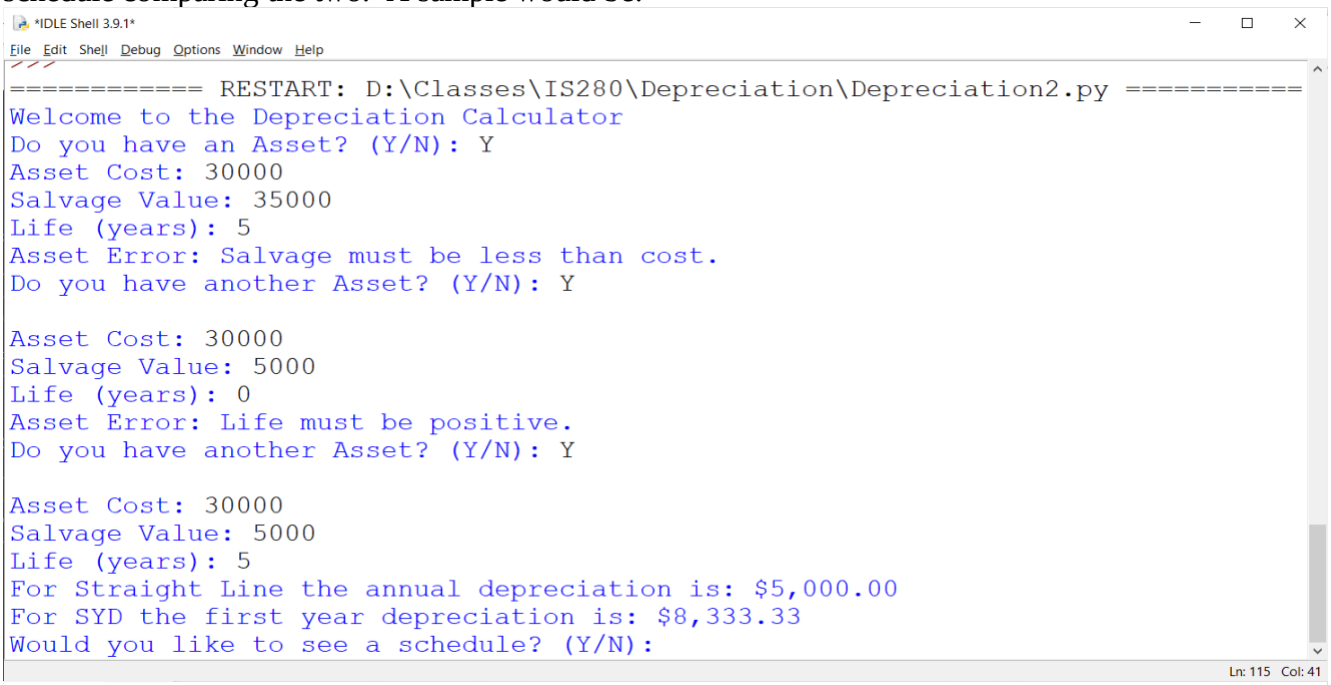
An example for a $100,000 asset with $10,000 salvage value and 5 years of useful life:

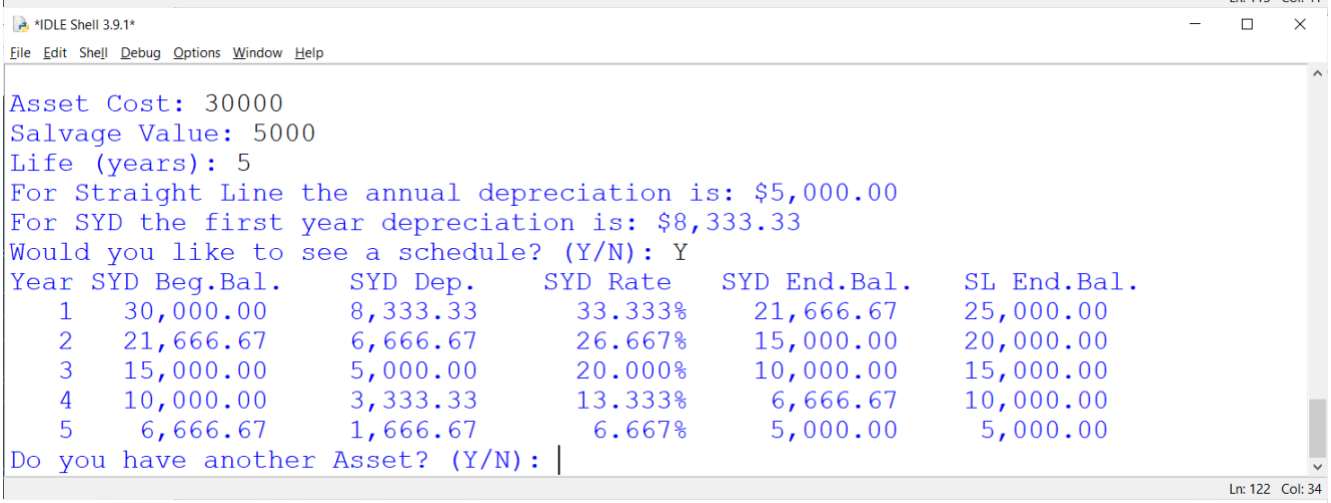
Year	*Remaining estimated useful life at beginning of year*	*SYD*	*Depreciation Rate*	*Annual depreciation*
1	5	5/15	33.33%	$30,000
2	4	4/15	26.67	24,000
3	3	3/15	20.00	18,000
4	2	2/15	13.33	12,000
5	1	1/15	6.67	6,000
Totals	15		100.00%	$90,000

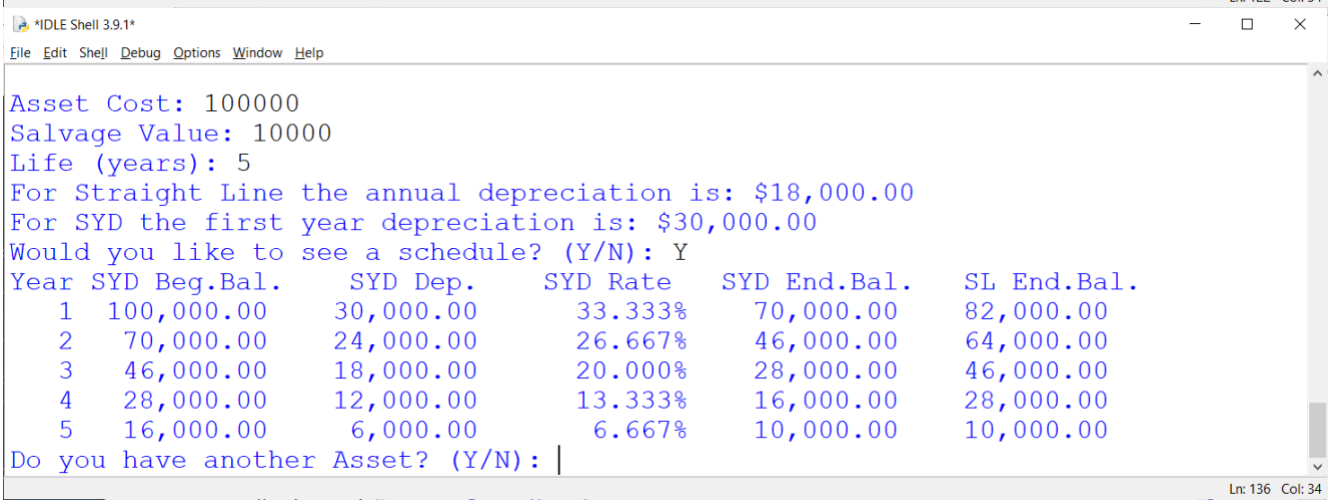
Note that there is a shortcut to calculating the sum of the year's digits (the denominator in the formula): **n(n+1)/2** where 'n' = number of years of useful life.

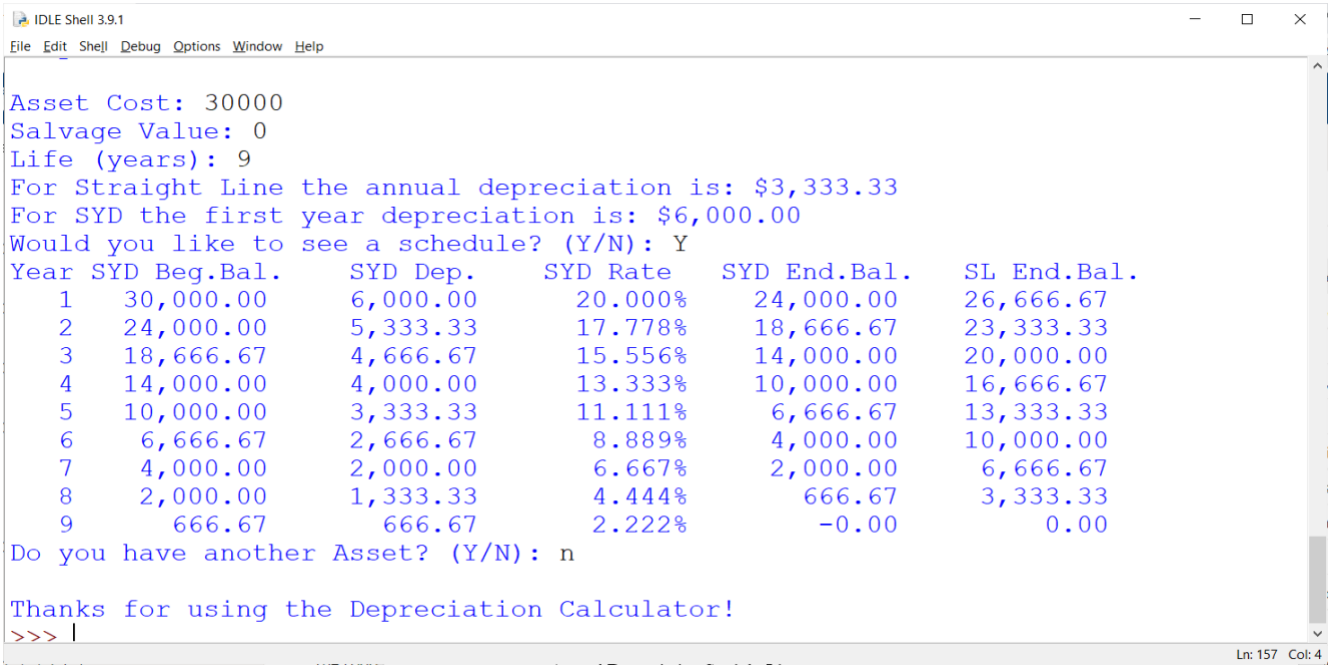
The values needed for the depreciation calculations are always: **Cost, Salvage Value, and Life of Item**

Develop a program to calculate both types of depreciation – Straight Line and SYD – and which can produce a schedule comparing the two. A sample would be:









(The SYD Annual rate shown above uses a {: .3%} type format control)

Build the application with a main Depreciation.py file to provide user I/O, control the process and provide the loop for multiple assets. Then use a separate Asset.py file to contain a class called Asset which will perform the required calculations. ***As we did with Annuity and Loan in the Financials project***, you should instantiate the Asset class by calling the __init__ method and should provide 'private' variable operation for the key asset values. Methods needed in the Asset class are:

- getCost() – to return the original asset cost
- getSalvage() – to return the original salvage value
- getLife() – to return the original asset life
- isValid() - the asset class should ***validate the starting values*** as follows:
    - 1) Cost and life must be greater than zero,
    - 2) Salvage value cannot be negative (but can be zero), and
    - 3) Salvage value must be less than cost.
    - Life of the asset is an integer; the other values are floats.
- getBegBal(yr) – to return the beginning balance for specified 'yr' for SYD schedule
- getAnnDepRate(yr) – to return the SYD depreciation rate for year 'yr'
- getEndBal(yr) – return SYD Ending Balance for year 'yr'
    - The Beginning Balance, Ending Balance, and Depreciation Rate should be carried as lists, created at instantiation
- getEndBalSL(yr) – return Ending Balance for Straight Line for year 'yr'. The return value can be calculated directly as needed:
    - End Bal for Year y = Asset Cost – (SL Annual Dep. * y)
- getAnnDep(yr – optional) is an overloaded method to return the annual depreciation:
    - **Special note on Annual Depreciation**: you should implement this method for getting the annual depreciation as **an overloaded method** (as we did in the Trivia Quiz program).
    - If a call to the get method does not include a year value (i.e., no parameter value is sent), then the method returns the Straight-Line annual depreciation.
    - If the call to the get method includes a year value, then the SYD Annual Depreciation for that year is returned.
    - Annual Depreciation for SYD should also be carried as a list, created at instantiation

Follow the exam instructions to re-open the test and post your two solution files: Depreciation.py and Asset.py.