Credit Score Estimator

Using Math and If-Statements

# Prior Required Knowledge

* Creating a project in PyCharm and running your code
* Variables and data types
* Printing information to the screen
* Converting types using str(), int(), and float()

# Problem Description

Many companies use credit scores to determine if they will approve customers for new loans or credit cards. For this project, you will write a program that will calculate a credit score for our company, ACME Inc. We wish to input data about our customers and determine if (and how much) we will loan an applicant. We calculate a score using the following:

|  |  |  |  |
| --- | --- | --- | --- |
| Category | Description | Criteria | Score |
| Payment History | Number of months since a negative report was made about the applicant. | No record of any  0 – 5  6 – 11  12 – 23  24+ | 75  10  15  25  55 |
| Outstanding Debt | Balance of debt. | No accounts  0  1 – 99  100 – 499  500 – 749  750 – 999  1000+ | 30  55  65  50  40  25  15 |
| Credit History Length | Number of months applicant has had credit. | <12  12 - 23  24 - 47  48+ | 12  35  60  75 |
| New Credit | Number of new credit attempts in the last 6 months. | 0  1  2  3  4+ | 70  60  45  25  20 |
| Credit Cards | Number of credit accounts that are credit cards. | 0  1  2  3  4+ | 15  25  50  65  50 |

**Overall Score = 500 + [Sum of the scores from each row in the table]**

Our approval criteria are as follows:

* No one approved for any loan with a score less than 650
* Approval for a loan $500 - $2,000 if no new credit in last six months and no record of missed payments in the last year, and score is over 650.
* Approval for a loan $500 - $5,000 if no new credit in last six months and no record of missed payments at all, and score is over 700.
* Approval for a loan $500 - $10,000 if score is over 800 and no record of missed payments.

# Program Design

Most programs are not composed of a large sequence of commands one right after the other. Typically, we take pieces of code that we wish to reuse and put them into smaller, named chunks called **functions**. We will be talking about functions soon, but for this project just write all your commands one after the other. Try to think about small sub-problems at a time – for instance first write the code for taking the input for Payment History and evaluating it; once you have that part working work on the next section. We haven't talked a lot about comments yet, so don't worry about commenting your code unless you are already familiar with comments. But – put extra lines between code sections. Be careful to save your work – MANY programmers have lost working code by messing around with it and trying new things. Be sure to keep a backup so this doesn't happen to you!

We will need to ask the user for input for this program to work. Python lets us ask a user to enter data with the input()function. This function allows us to provide an optional prompt. The function will return the user's input *as a string*. This means that even though you ask for a number, whatever the user enters will be a string. We can try to convert the string to a number using the int()or float() functions. For example:

>>> num = input("What is your number? ")

What is your number? 42

>>> type(num)

<class 'str'>

>>>

>>> int(num)

42

>>> float(num)

42.0

>>>

Here we asked a user for a number and stored it in a variable called "num". If we hadn't set num equal to what input() returned we would have lost the user's input. Since we stored it in a variable, we can now work with that data. I first checked to see what type of data the num variable was holding – as the type() function shows num is holding a string. I use the int() and float() functions to see what value num would be if interpreted as an integer or a floating point number (a number with a decimal portion like 3.14159). Notice that those two functions didn't change the type of data for the num variable, they simply showed the value as an integer or a float. If I wanted to store the integer or float value, we would need to assign the value to a variable.

# Grading

I will use the following rubric to grade your project:

* Code for each of the five sections (i.e., Payment History, Outstanding Debt, etc.) is correct and adding the correct amount to the overall score. **10 points per section.**
* Neatly formatted code with lines between program "sections" (as described in the Program Design section above) **10 points.**
* Working, complete in-lab demonstration. **10 points.**