NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Life Contingencies I

Academic Year 2023/2024

Practical Assignment

Francisco Brito, Nº 68769

Sebastian Vraja, Nº 67110

Professor Doctor Pedro Alexandre da Rosa Corte Real

DM FCT-NOVA

December 2023

UNIVERSIDADE NOVA DE LISBOA

Índice

1 Introduction

Life insurance is an important and fundamental aspect of financial planning, providing individuals with means to safeguard their loved ones against the uncertainty of life. In the context of Life Contingencies I course, we have developed a user-friendly application, that allows both the insured person and the life actuary to make informed decisions regarding a **term life insurance product**, where the euro €, is the based monetary unit used. This program utilizes a **graphical user interface (GUI) built with Tkinter in *Python***, offering an easy and straightforward experience to both parties involved, allied with the use of *pandas* and *numpy* libraries offering optimized numerical calculations.
The program accommodates various functionalities to allow for the needs of both the insured individual and the Life Actuary. Users can input essential details, such as the insured person's age, desired insurance term - for the term life insurance product, and the period over which leveled premiums will be paid, typically these 3 details are up to the insured person to decide upon. Moreover, the program allows the selection of a flat interest rate, a life table (***GRM95*, *GRF95* or *TV95***), the classification of the insured life (***Preferred* , *Standard* , *Aggravated***), depending on the results of mandatory medical exam, and the portfolio size ($\mathbf{L_0}$). This last few functionalities are exclusively intrinsic to the Life Actuary to decide upon.
Through this intuitive GUI, the program provides insightful outputs, including the **calculation of the single risk premium, leveled premiums, the expected fund path for the portfolio, considering premiums received and claims paid**. It also offers the ability to see **graphs with premiums received, claims paid, and the population sizes through time**, together with this, also offers a **short report** for better visualization of the most important elements to help the insured person make an informative decision. All of this functionalities are in a distance of a button.
In the subsequent sections of this report, we will indulge into the specification of the code produced, showing the underlying logic of the decisions made and detailed program's functionalities. Furthermore, there is a section with practical examples to guide users in producing results, alongside a discussion on both the strengths and potential limitations of the project developed. This report closes with the conclusion section, where the main concepts and information is wrapped and summarised.

2 Code Overview and Guidance with Examples

In this section, information is organized in a straightforward manner, to serve as guide or information source material for the program developed, with selected examples and their expected results for reference.

2.1 User Interface

The graphical user interface **(GUI)** of this Term Life Insurance program is designed to provide a user-friendly experience for configuring and obtaining Term Life Insurance related calculations, useful for both the insured person/person looking to buy an Term Life Insurance product, and the life Actuary. The user interface comprises of several input fields and buttons, each serving a specific purpose.

Input fields and Labels

Insured Person's Details:

- **Insured Age(x):** A spinbox allowing the user to input the age of the insured person. The age should be between 18 and 65 years.
- **Term of Life Insurance(n):** Another spinbox allowing the user to specify the term for life insurance. This ranges from 5 to 25 years.
- **Leveled Premium Period (n_lev):** Another spinbox to set the period for paying leveled premiums, which can be selected between 5 and 25 years.

Life Actuary's Inputs:

- **Flat Interest Rate (tx):** A spinbox to input the flat interest rate, ranging from 1% to 5%.
- **Underwritten Capital ($Capital$):** A spinbox allowing the user to set the underwritten capital for the insurance.
- **Initial Size of the Population (L_0):** A spinbox for specifying the number of people alive at age 0.
- **Select a table:** Radio buttons ($GRM95$, $GRF95$, $TV95$) for the life actuary to choose the life table to be used in calculations.

- **Life Insured Type:** Radio buttons(*Aggravated*, *Standard*, *Preferred*) for classifying the health condition of the insured life. Life Actuary should consult with the medical team to ensure correct option is being selected.

Buttons

- **Risk Single Premium:** A button trigerring the calculation and display of the risk single premium based on the inputs, provided on a different window.
- **Table with Leveled Premiums:** A button used to generate and display in a different window, a table showing the desired information such as age, population size, premiums, claims, fund value and net premium reserve. Also, below this table, the annuity used to calculate the leveled premium is displayed.
- **Graphs:** A button that creates in a different window, visual representations of premiums vs age, claims vs age, and population size vs age.
- **Summary Report:** A button generating a concise report summarizing the conditions and expected values for the selected insurance parameters. Provides guidance to the Insured Person to help her/him make a decision.

2.2 Code Explanation

This section aims to provide further information on how each function was created in the code and how everything interacts. In some instances there is mathematical reference to how each function was derived. For further enquire on the programming language technicalities, consulting the commented code delivered alongside this report, is advised.

1. `Table(var1, var2, x)`

This function is responsible for selecting and preparing a life table based on the life actuary's choices (`var1` - Table choice input and `var2` - Insured type input). It reads a CSV file (`GRM95.csv`, `GRF95.csv`, or `TV7377.csv`), extracts the necessary columns, and renames them to "Age" and "qx." The function also adjusts mortality rates (q_x) based on the life insured type (`var2`). The resulting DataFrame is returned.

2. `npx(n, x, Table)`

Calculates the probability function ${}_np_x$, representing the probability of survival for a given age x over n years using the provided life table (`Table`). It retrieves the relevant subset of the table and calculates the product of survival probabilities. Note that due to data availability and to try as much not to derive ${}_np_x$ as a function of l_x, the probability function was created based on the following mathematical rationale:

$$
\begin{aligned}
{}_np_x &= \frac{l_{x+n}}{l_x} \\
&= \frac{l_{x+n-1} \cdot p_{x+(n-1)}}{l_x} \\
&= \frac{l_{x+n-2} \cdot p_{x+n-2} \cdot p_{x+(n-1)}}{l_x} \\
&\vdots \\
&= \frac{l_x \cdot p_x \cdot p_{x+1} \cdot \ldots \cdot p_{x+(n-1)}}{l_x} \\
&= \prod_{i=0}^{n-1} p_{x+i}
\end{aligned}
$$

Please note that in order to get p_x values we derive $p_x = 1 - q_x$, which q_x are already calculated depending on the classification of the life by the Life Actuary and the medical team. Having into account a selection period of $d = 3years$, please note the given following rationale:

Aggravated: $q_{[x]} = 1,20 \cdot q_x$, $q_{[x-1]+1} = 1,10 \cdot q_x$, $q_{[x-2]+2} = 1,05 \cdot q_x$

Preferred: $q_{[x]} = 0,80 \cdot q_x$, $q_{[x-1]+1} = 0,90 \cdot q_x$, $q_{[x-2]+2} = 0,95 \cdot q_x$

It's implicit that standard Life classification is not going to change the values taken directly from the Life Table selected.

3. `A(x, n, i, Table)`

The term life insurance policy used for this function was the annual case, therefore this function computes the present value of a death benefit of 1 that is payable at the end of the year of death, provided this occurs within `n` years ($A_{x:\overline{n}|}$). It iterates through each year, calculating the discounted value of the death benefit using the interest rate i ($i = tx$) and the relevant survival and death probabilities from the life table(`Table`).

$$A^1_{x:\overline{n}|} = \sum_{j=0}^{n-1} v^{j+1} \cdot {}_jp_x \cdot q_{x+j}$$

4. `risksingle(var1, var2, x, n, tx, capital)`

Calculates the risk single premium based on the life actuary's choices and user inputs. already identified previously. It calls the `Table` function to get the relevant life table and uses the `A` function to determine the present value of death benefits. The final premium is computed by multiplying the death benefit(`A`) by the underwritten capital($Capital$). Note the used mathematical formula in this function:

$$P = Capital \cdot A^{1}_{x:\overline{n}|}$$

5. `axn(x, n_lev, i, table)`

Calculates the annuity used to level the premium payments over n_lev years ($\ddot{a}_{x:\overline{n}|}$). It iterates through each year, computing the present value of the survival probabilities using the interest rate `i`.This function follows the used actuarial mathematical notation for annuity:

$$\ddot{a}_{x:\overline{n}|} = \sum_{j=0}^{n-1} v^{j+1} \cdot {}_{j}p_{x}$$

6. `lev_premium(x, n, n_lev, capital, i, table)`

Determines the leveled premium by dividing the present value of death benefits (`A`) by the annuity used to level the premium payments (`axn`).

7. `kVxm(k, x, n, n_lev, i, table)`

Computes the net premium reserve for a specific year `k`. It involves calculating the present value of future death benefits and subtracting the present value of future premiums.This calculation is typically used in Insurance companies in order to "reserve" sufficient funds to cover for this liability. This function was built following the standard actuarial formula and uses already defined function and user inputs.

For $k < n$,

$$V^{\,1}_{x+k:\overline{m-k}|} = A^{\,1}_{x+k:\overline{m-k}|} - \frac{A^{1}_{x:\overline{m}|}}{\ddot{a}_{x:\overline{n}|}}$$

For $k > n$,

$$V^{\,1}_{x+k:\overline{m-k}|} = A^{\,1}_{x+k:\overline{m-k}|}$$

8. `tdf(x, n, l_0, n_lev, tx, capital, var1, var2)`

This is where the magic happens.Generates the target DataFrame which includes age, population size (`lx`), premiums, claims, fund, and net premium reserves. It utilizes the `Table` function to obtain the life table and calculates intermediate values for premiums and reserves. Please note the below table, displaying what was the rationale to derive each column. Due to simplification, the notation showed is the generic one.

Age (x)	**Population Size** (l_x)	**Premium (4)**	**Claims (5)**	**Fund Value (4)+(6)**	**Net Premium Reserve (6)**
x	l_x	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_x$	0	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_x + {}_1V^1_{x:\overline{n}\rvert}$	${}_1V^1_{x:\overline{n}\rvert}$
$x+1$	l_{x+1}	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_{x+1}$	$C \cdot [l_{x-1} - l_x]$	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_{x+1} + {}_2V^1_{x:\overline{n}\rvert}$	${}_2V^1_{x:\overline{n}\rvert}$
$x+2$	l_{x+2}	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_{x+2}$	$C \cdot [l_{x-1} - l_x]$	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_{x+2} + {}_3V^1_{x:\overline{n}\rvert}$	${}_3V^1_{x:\overline{n}\rvert}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$x+n$	l_{x+n}	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_{x+n}$	$C \cdot [l_{x-1} - l_x]$	$\frac{C \cdot A^1_{x:\overline{n}\rvert}}{\ddot{a}_{x:\overline{n}\rvert}} \cdot l_{x+n} + {}_nV^1_{x:\overline{n}\rvert}$	${}_nV^1_{x:\overline{n}\rvert}$

9. `display_df(x, n, l_0, n_lev, tx, capital, var1, var2)`

Displays the DataFrame generated by the `tdf` function in a new Tkinter window. It also calculates and displays the annuity used to level the premiums. For futher reference, note section **2.3** where examples for the displayed Data Frame are showed.

10. `graph_df(x, n, l_0, n_lev, tx, capital, var1, var2)`

Generates a new Tkinter window with three subplots representing Premiums vs. Age, Claims vs. Age, and Population Size vs. Age using the DataFrame created by `tdf`.

11. `window_risk(var1, var2, x, n, tx, capital)`

Creates a new window displaying the calculated risk single premium using the `risksingle` function.

12. `last_report(x, n, n_lev, tx, capital, var1, var2)`

Generates a summary report in a new *Tkinter* window, providing key information such as insured age, term, leveled premium period, underwritten capital, and flat interest rate, along with calculated single premium and annual leveled premium.

2.3 Examples

In this section, two examples were provided to demonstrate the functionality of this Term Life Insurance program. For each example, a set of insured and Life Actuary inputs are used and visual representation of the expected outputs and outcomes of program's functionalities, these are the outputs that come directly from the interactive **Buttons** already defined.

Example 1

Inputs EX1

- **Insured Person's input**
  - Insured Age: $30 years$
  - Term of Life Insurance: 10
  - Leveled Premium Period: 3
- **Life Actuary's input**
  - Flat Interest Rate: 3%
  - Underwritten Capital: 1000
  - Initial Size of Population: 100000
  - Select a table: $GRM95$
  - Life Insured Type: $Standard$

Expected Output EX1

- **Risk Single Premium:**

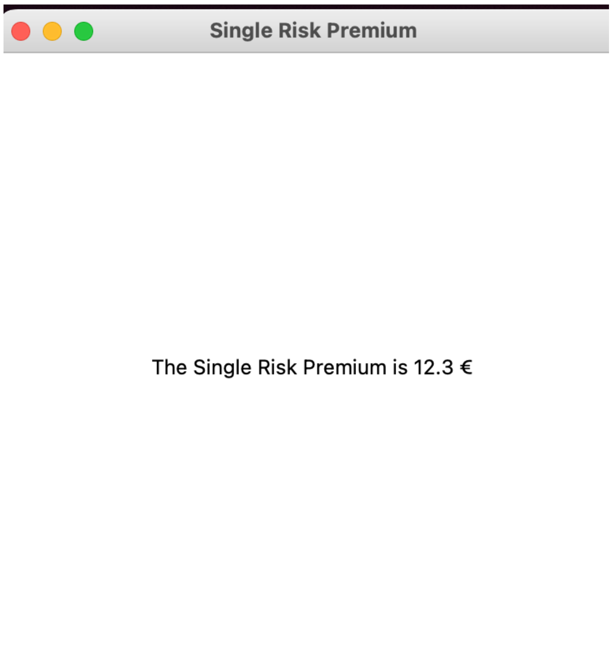


Figure 1: Risk Single Premium

- **Table with Leveled Premiums:**

Leveled Premiums

Age	lx	Premium	Claims	Fund	Net Premiums Reserve
30.0	100000.0	422772.82	0.0	422772.82	0.0
31.0	99869.43	422220.81	130570.0	727106.82	304886.01
32.0	99738.14	421665.76	131288.35	1039297.43	617631.67
33.0	99605.24	0.0	132901.07	937575.28	937575.28
34.0	99469.82	0.0	135423.29	830279.25	830279.25
35.0	99330.91	0.0	138909.6	716278.03	716278.03
36.0	99187.51	0.0	143394.1	594372.27	594372.27
37.0	99038.59	0.0	148920.13	463283.3	463283.3
38.0	98883.0	0.0	155589.63	321592.17	321592.17
39.0	98719.5	0.0	163503.05	167736.89	167736.89
40.0	98546.73	0.0	172769.0	0.0	0.0

The Annuity used to level the premiums is 1.3120099135295764

Figure 2: Target Table with Leveled Premiums and annuity used,claims,fund,Net Premium Reserves calculations.

- **Graphs:**

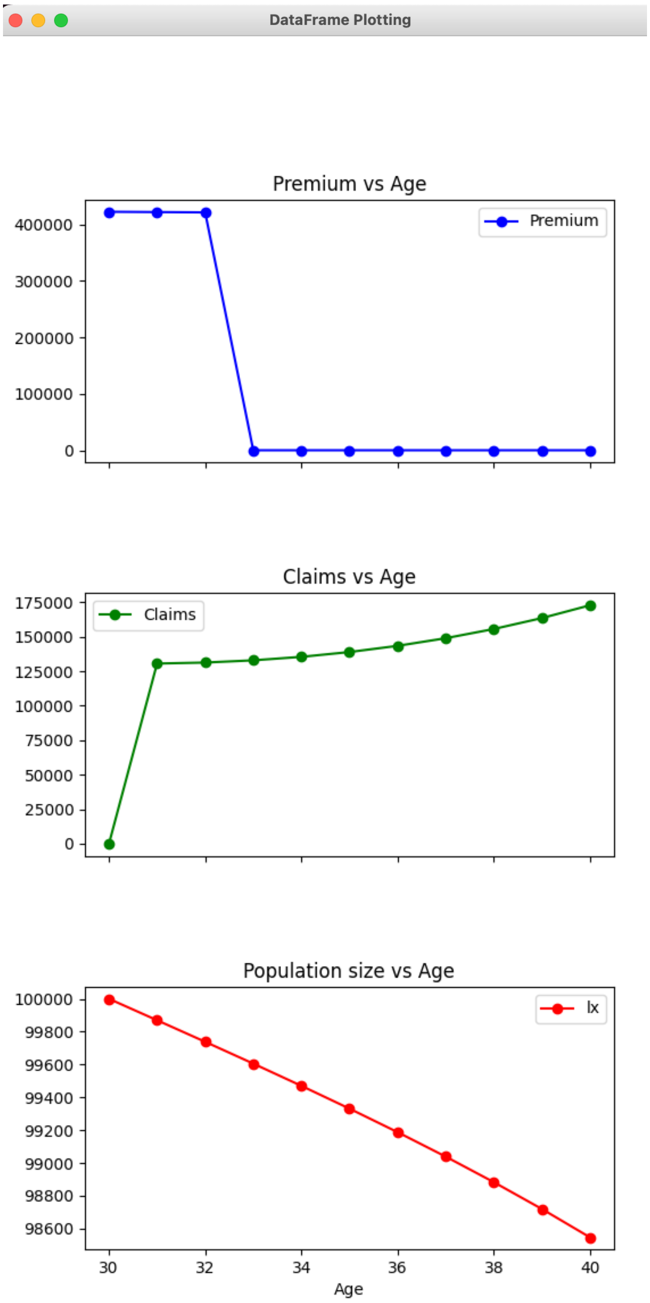
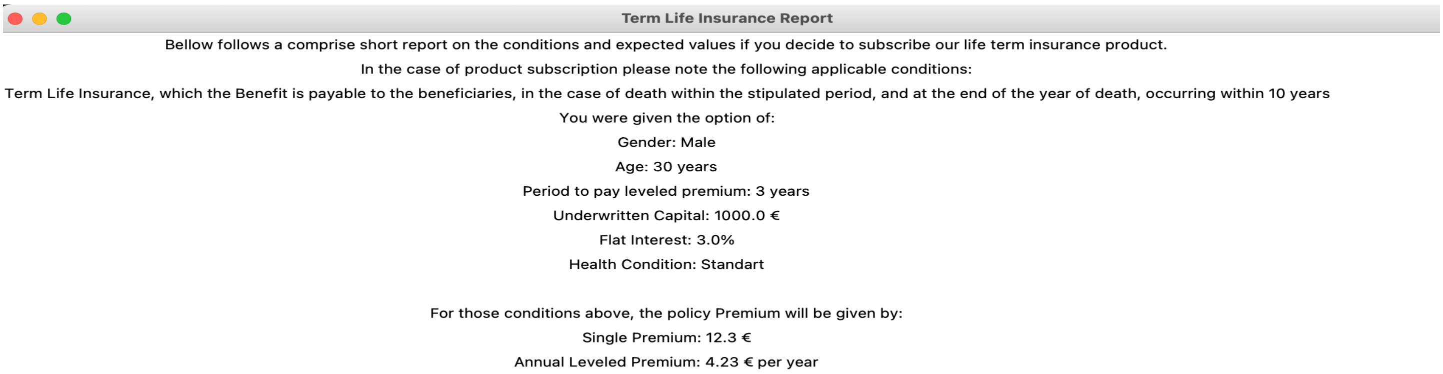


Figure 3: Graphs

- **Summary Report:**



Term Life Insurance Report

Bellow follows a comprise short report on the conditions and expected values if you decide to subscribe our life term insurance product.

In the case of product subscription please note the following applicable conditions:

Term Life Insurance, which the Benefit is payable to the beneficiaries, in the case of death within the stipulated period, and at the end of the year of death, occurring within 10 years

You were given the option of:

Gender: Male

Age: 30 years

Period to pay leveled premium: 3 years

Underwritten Capital: 1000.0 €

Flat Interest: 3.0%

Health Condition: Standart

For those conditions above, the policy Premium will be given by:

Single Premium: 12.3 €

Annual Leveled Premium: 4.23 € per year

Figure 4: Summary Report

Example 2

Inputs EX2

- **Insured Person's input**
  - Insured Age: 45$years$
  - Term of Life Insurance: 5
  - Leveled Premium Period: 3
- **Life Actuary's input**
  - Flat Interest Rate: 4%
  - Underwritten Capital: 1000
  - Initial Size of Population: 100000
  - Select a table: $GRF95$
  - Life Insured Type: *Aggravated*

Expected Outputs EX2

- Risk Single Premium:

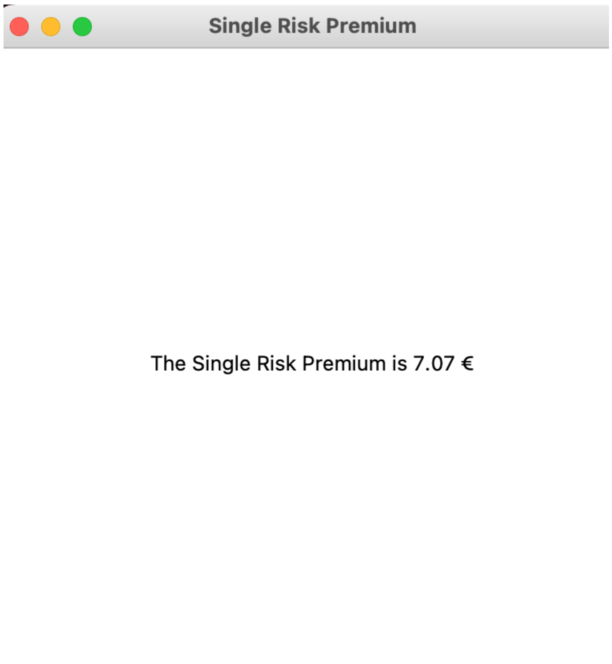
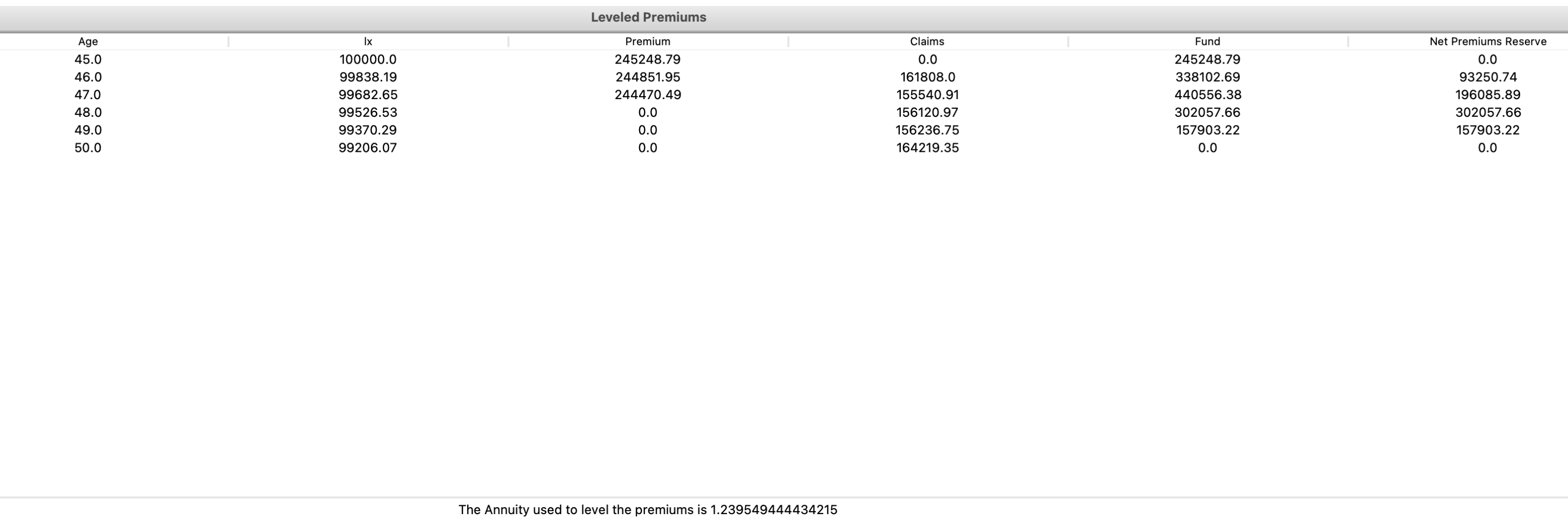


Figure 5: Risk Single Premium

- Table with Leveled Premiums:



Leveled Premiums

Age	lx	Premium	Claims	Fund	Net Premiums Reserve
45.0	100000.0	245248.79	0.0	245248.79	0.0
46.0	99838.19	244851.95	161808.0	338102.69	93250.74
47.0	99682.65	244470.49	155540.91	440556.38	196085.89
48.0	99526.53	0.0	156120.97	302057.66	302057.66
49.0	99370.29	0.0	156236.75	157903.22	157903.22
50.0	99206.07	0.0	164219.35	0.0	0.0

The Annuity used to level the premiums is 1.239549444434215

Figure 6: Target Table with Leveled Premiums and annuity used,claims,fund,Net Premium Reserves calculations.

- Graphs:

Figure 7: Graphs

- Summary Report:

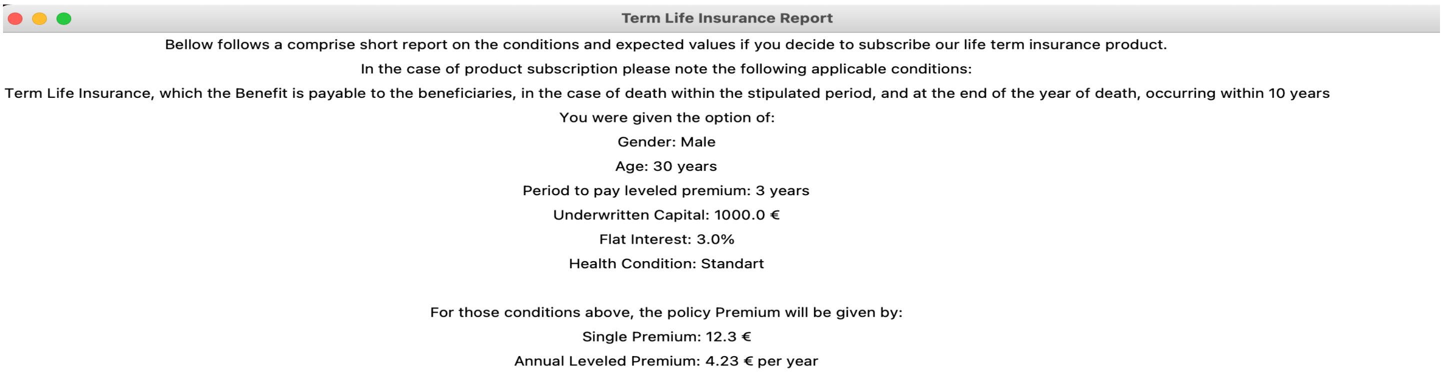


Figure 8: Summary Report

2.4 Strengths and Weaknesses

Strengths of the Program

The tkinter-based interface ensures a user-friendly experience, making it accessible for both insured individuals and life actuaries. Adding to this, both Insured and Life actuaries can choose from different life tables, providing adaptability to varying scenarios and requirements.The program generates a comprehensive output, including risk single premiums, leveled premiums, and graphical representations, offering a holistic view of the insurance evaluation.

Weaknesses of the Program

The program lacks extensive error handling mechanisms, potentially leading to unexpected behavior when incorrect inputs are provided. There is some degree of numerical imprecision, specially in the computation of the `tdf` function, which could affect results accuracy. Also, no warning messages are displayed when an input is invalid. From an actuarial point of view, there is a lack of functionalities as this application doesn't allow for the death benefit to be immediately paid in case of death, nor to be paid in the end of the death month. Overall there are a couple of extra features that could be added to the program, allied with more information from the medical team more advanced details could be added in and serve as input. There is definitely room for improvement and should serve as base for future development and studies.

3 Conclusion

In conclusion, this Term Life Insurance project provided valuable insights into both actuarial principles and computational techniques. From an actuarial standpoint, we gained a deeper understanding of risk assessment, premium calculations, and the importance of life tables. The program's implementation in Python, coupled with a user-friendly interface using Tkinter, showcased the practical application of computational tools in the actuarial field. The project highlighted the significance of integrating mathematical concepts with programming skills, contributing to a holistic learning experience in both actuarial science and computational methods.