University of Thessaly – Laboratory of Biomathematics
MSc Research Methodology in Biomedicine, Biostatistics and Clinical Bioinformatics

Software development in Python for a 3:3:2 (Test drug: Reference drug: Placebo) randomization in clinical trials.

Ανάπυξη λογισμικού σε Python για 3:3:2 (Νέο Φάρμακο: Φάρμακο Σύγκρισης: Εικονικό Φάρμακο) τυχαιοποίηση σε κλινικές δοκιμές.

Ermioni Kritsinioti – Ερμιόνη Κριτσινιώτη 0162

Επιβλέπων - Supervisor:

Kowald Axel

Τριμελή Επιτροπή Αξιολόγησης - Three-member Committee :

Kowald Axel

Χρυσούλα Δοξάνη

Ηλίας Ζιντζαράς

21-Sep-18

Software development in Python for a 3:3:2 (Test drug: Reference drug: Placebo) randomization in clinical trials.

Table of Contents

Software development in Python for a 3:3:2 (Test drug: Reference drug: Placebo) randomization in clinical trials.

Περίληψη

Η κλινική δοκιμή/μελέτη διεξάγεται σε εθελοντές ώστε να αποφασιστεί εάν ένα νέο φάρμακο είναι ασφαλές και αποτελεσματικό στους ανθρώπους. Οι κλινικές μελέτες κατατάσσονται σε τέσσερις φάσεις, οι οποίες θα επεξηγηθούν.

Μια σημαντική πτυχή στην εκπόνηση των επιστημονικών μελετών είναι η τυχαιοποίηση, δηλαδή η κατανομή των συμμετεχόντων στα διάφορα σκέλη της μελέτης ώστε να οφείλονται αποκλειστικά στην τύχη.

Η τεχνική της τυχαιοποίησης προλαμβάνει την μελέτη απο συστηματικά σφάλματα επιλογής (selection bias) και συστηματικά σφάλματα συγχυτικών παραγόντων (confounding bias).

Ο κύριος στόχος της παρούσας εργασίας είναι η ανάπτυξη λογισμικού που παράγει λίστες υποψηφίων(ασθενών) σε κλινικές μελέτες φάσης ΙΙ, ΙΙΙ και συγκεκριμένα μελέτες που εξετάζουν την ανταπόκριση ενός νέου φαρμάκου(test drug), φαρμάκου σύγκρισης(reference drug) και εικονικού φαρμάκου(placebo), με αναλογία κατανομής 3:3:2. Ο χρήστης μπορεί να επιλέξει ανάμεσα στους 3 τύπους τυχαιοποίησης(Απλή, Μπλοκ, Στρωματοποιημένη)

Το εργαλείο τυχαιοποίησης αναπτύχθηκε μέσω του Eclipse IDE σε Python 3 γλώσσα. Επιπλέον, ανέπτυξα ένα διαδραστικό γραφικό περιβάλλον για τον χρήστη με την υποστήριξη της βασικής βιβλιοθήκης Tkinter.

Παράλληλα με την ανάπτυξη της εφαρμογής, η πτυχιακή αυτή περιλαμβάνει μια θεωρητική παρουσίαση ορισμένων μεθόδων, διαδικασιών και λειτουργιών παραγωγής λογισμικού. Επιπλέον, ένα ακόμη θεωρητικό στοιχείο που αναλύεται στην εργασία αυτή, είναι τα οφέλη που προκύπτουν από τις τυφλές κλινικές δοκιμές και κατ' επέκταση η χρηστικότητα του συγκεκριμένου εργαλείου τυχαιοποίησης.

Το εργαλείο αυτό έχει πολλές δυνατότητες, ωστόσο υπάρχει χώρος για βελτιώσεις. Μία επιπλέον λειτουργία που θα μπορούσε να υλοποιηθεί είναι να παρέχει τη δυνατότητα επιλογής επιθυμητών αναλογιών κατανομής και διαφορετικών θεραπευτικών ομάδων.

Abstract

Clinical trials are conducted on volunteers to decide whether a new drug is safe and effective for humans. Clinical trials are divided into four phases, which will be shortly explained in the introduction.

An important aspect in the development of scientific studies is randomization, in which the participants are divided by chance into separate groups that compare different treatments. Randomization prevents the study from systematic selection bias and systematic confounding bias. There are three Randomization types (Simple, Block, Stratified), which will be explained.

The main objective of this work is to develop software that produces patients' candidate lists in for mainly Phase II, III clinical trials, which study the response of a new drug (test drug), a reference drug and a placebo, with a distribution ratio of 3: 3: 2. There will be available 3 options for the user, which are Simple, Block and Stratified Randomization.

The randomization tool was developed through Eclipse IDE in Python 3 language. I developed an interactive graphical user interface (GUI) with the support of the built in Tkinter library.

Along with the development of the application, this project includes a theoretical presentation of the software production methods, procedures and functions. In addition, another theoretical element analyzed in this work, are the benefits of a blinded clinical trial and, by extension, the usability of this randomization tool.

This tool has functional features; however there is room for improvement. An additional function that could be implemented is to provide the choice of desirable distribution ratios and different therapeutic groups to the user.

2. Introduction

Clinical Trials

Clinical trials are conducted on volunteers to decide whether a new drug is safe and effective for humans. These studies often help scientists to understand in depth pathophysiological mechanisms of the disease. Clinical trials are divided into four phases: (Novartis, 2017)

• Phase I, an experimental treatment is first studied in a small group of individuals (20- 80) to evaluate how the body absorbs, distributes, metabolizes and removes the drug in order to identify the optimal route of administration and dose (the individuals participating in the clinical trial study are usually healthy volunteers)

• Phase II, experimental therapy is given to a larger group of patients (100-300) in order to check the efficacy and further assess the safety of use (the patients in the clinical trial are patients with the relevant disease)

• Phase III, experimental therapy(test drug) is administered to much larger groups (1.000-3.000) to confirm its efficacy, follow the side effects, and compare it to the reference drug, the frequently used treatments or the placebo (if positively evaluated by the approval authorities, it then receives the necessary marketing authorization)

• Phase IV, involving formulations that have already been released and included in the clinical trial, additional information is provided from their daily clinical application, in particular the benefits and side effects in order to confirm their best use.

Blinded Clinical Trials

Human behavior is influenced by what we know or believe. In research there is a particular risk of expectation influencing findings, most obviously when there is some subjectivity in assessment, leading to biased results. Blinding (sometimes called masking) is used to try to eliminate such bias. It is a tenet of randomized controlled trials that the treatment allocation for each patient is not revealed until the patient has irrevocably been entered into the trial, to avoid selection bias. In a double blind trial neither the patient nor the caregivers are aware of the treatment assignment.

If we want to compare two medicines, one presented as green tablets and one as pink capsules, we could also supply green placebo tablets and pink placebo capsules so that both groups of patients would take one green tablet and one pink capsule. (Day SJ, 2000)

Therefore, the Randomization tool developed for this work offers benefits of a blinded clinical trial by conceiving the randomization process and allocating patients as hidden.

Randomization

An important aspect in the development of scientific studies is randomization. Randomization is the distribution of the participants in the various strands of the study in such a way that the differences between the strands of the study are due solely to chance. Each patient has the same chance of getting one or the other treatment. (Novartis, 2017)

The randomization technique protects the reliability of the results of each study: (Suresh, 2011)

- It prevents the selection bias and insures against the accidental bias (systematic selection bias and systematic confounding bias)
- It produces the comparable groups and eliminates the source of bias in treatment assignments.
- It permits the use of probability theory to express the likelihood of chance as a source for the difference of end outcome

Selection bias is the bias introduced by the selection of individuals, groups or data for analysis in such a way that proper randomization is not achieved, thereby ensuring that the sample obtained is not representative of the population intended to be analyzed. (wikipedia, n.d.)

Confounding bias is a situation in which the effect or association between an exposure and outcome is distorted by the presence of another variable. (Pennsylvania State University, n.d.)

There are three Randomization types: Simple, Block and Stratified.

Unbalanced Randomization

If a clinical trial compares a new therapy with standard therapy, we may be interested in gaining more experience in new treatment. Thus, we should assign more patients to the new treatment, although we may lose some statistical effectiveness. Unbalanced randomization is quite popular in Phase II, Phase III trials where there is no prior knowledge of the effectiveness of the new treatment or there is a need to compare two drug treatment groups with a placebo. (Altman, 1991, p. 90)

Therefore, a 3:3:2 ratio provides a larger representation of the new test drug and gives data for a drug that has never been used before, in contrast to placebo drugs.

Simple Randomization

Randomization based on a single sequence of random assignments is known as simple randomization. This technique maintains complete randomness of the assignment of a subject to a particular group. The most common and basic method of simple randomization is flipping a coin. For example, with two treatment groups (control versus treatment), the side of the coin (i.e., heads - control, tails - treatment) determines the assignment of each subject. (Suresh, 2011)

However, it should be mentioned that Simple randomization is most effective in large sample sizes under large scale clinical research.

In particular, for the software developed for this project, Simple Randomization takes as input a candidate number which is a multiple of 8, in order to keep the 3:3:2 ratio.

Block Randomization

The block randomization method is designed to randomize subjects into groups that result in equal sample sizes. This method is used to ensure a balance in sample size across groups over time (Suresh, 2011). Randomized blocks can be of any Size, but using a multiple of the number of treatments is more logical and is desirable to vary the block length at random. The block size is randomly assigned and should be a multiple of the number of groups (i.e., with three treatment groups and equal possibility, block size of either 3, 6, or 9) (Altman, 1991).

After block size has been determined, all possible balanced combinations of assignment within the block (i.e., equal number for all groups within the block) must be calculated. Blocks are then randomly chosen to determine the patients' assignment into the groups. (Suresh, 2011)

In terms of this software, a block size of 8 is used in order to keep the desired ratio. If the sample size is very small block and stratified randomization cannot be used.

Stratified Randomization

The stratified randomization method addresses the need to control and balance the influence of covariates (Suresh, 2011). We can use stratified randomization to achieve approximate balance of important characteristics without sacrificing the advantages of randomization. The method is to produce a separate block randomization list for each subgroup (stratum) (Altman, 1991). Specific covariates must be identified by the researcher who understands the potential influence each covariate has on the dependent variable.

Stratified randomization is achieved by generating a separate block for each combination of covariates, and subjects are assigned to the appropriate block of covariates. After all subjects have been identified and assigned into blocks, simple randomization is performed within each block to assign subjects to one of the groups (Suresh, 2011).

Stratified randomization is delivered in this software by asking the candidate number and the factors (covariates) that are important to this research. Note that in order to perform stratification efficiently, user is asked to enter a number that is multiple to 32(32, 64 etc.) and the number of candidates need to be divided by the strata as to fill at least one block of size 8. Strata are produced by combining each covariate's subgroup:

F.e if users need 2 covariates/factors like ‘Age: under 30 years, over 30 years’ and ‘Weight: Under 60 kgs, over 60kgs’, our stratas will equal to 2x2 = 4 stratifications.

Therefore Strata 1 is under 30 years/under 60kgs, strata 2 is under 30 years/over 60kgs, strata 3 is over 30 years/over 60 kgs and strata 4 is over 30 years /over 60kgs.

3. Methods

Software Development Life Cycle (SDLC)

The Software Production Process or Development Cycle Software Development Life Cycle is a process that ensures that the software is produced satisfies the requirements. It defines which tasks must be performed at each step in the software development process. There are usually 6 stages, beginning analysis and collection of requirements and ends with execution. (Hussung, 2016)

1. Planning “What don’t we want?”
2. Analysis. “What do we want?”
3. Design. “How will we get what we want?”
4. Implementation. “Let’s create what we want.”
5. Testing. “Did we get what we want?”
6. Deploy. “Let’s start using what we got.”
7. Maintain. “Let’s get this closer to what we want.”

In order to develop the randomization tool correctly, I followed the steps above that are included in every software program.

Randomize Application Development

Firstly, understanding the theoretical knowledge of the three randomization methods was critical for the development of the program. I used Python 3 (Python.org, 2008)in the Eclipse IDE (eclipse.org, 2018) to develop the code.

After that, I analyzed a rough algorithm plan in order to have the outline of what it needed to be implemented first. In order to guarantee the random allocation each time, I used random.seed(). Each randomization is developed in different functions:

- Simple randomization: Using random.shuffle() for the list of population/candidates , a list of random assignments is generated keeping the 3:3:2 ratio for each treatment group.

```
def create_windowA(self): '''opens simple randomization window'''
def simpleRand(self): '''calls Simple() and writeFileSimple()'''
```

```
def Simple(self,n): '''performs simple randomization'''
def writeFileSimple(self,results): '''outputs results in.txt file'''
```

- Block randomization: With Block size 8 for every block this function calls the Simple() randomization function for every block. Thus creating a list with blocks of treatment groups.

```
def create_windowB(self): '''opens block randomization window'''
def blockRand(self): '''calls Blocked() and writeFileBlock()'''
def Blocked(self,n): '''performs block randomization'''
def writeFileBlock(self,results): '''outputs results in.txt file'''
```

- Stratified randomization: This algorithm, uses the factors input from user to calculate the strata. For each combination of the covariates(strata) the Blocked() function is called and each strata has different blocks of randomly assigned patients in treatment groups.

```
def create_windowC(self): '''opens stratification randomization window'''
def getFactors(self): '''gets factors from user and calls strataRand()'''
def strataRand(self): '''finds strata and calls writeFileBlock()'''
def writeFileStrata(self,results,factors,levels): outputs stratification results'''
```

Error Handling

In continuation, the program was tested under different inputs and I corrected any rules and limitations that were against the theory of randomization. Thus, the efficiency of the code was tested successfully. The error codes of the rules and limitations are included in the functions below:

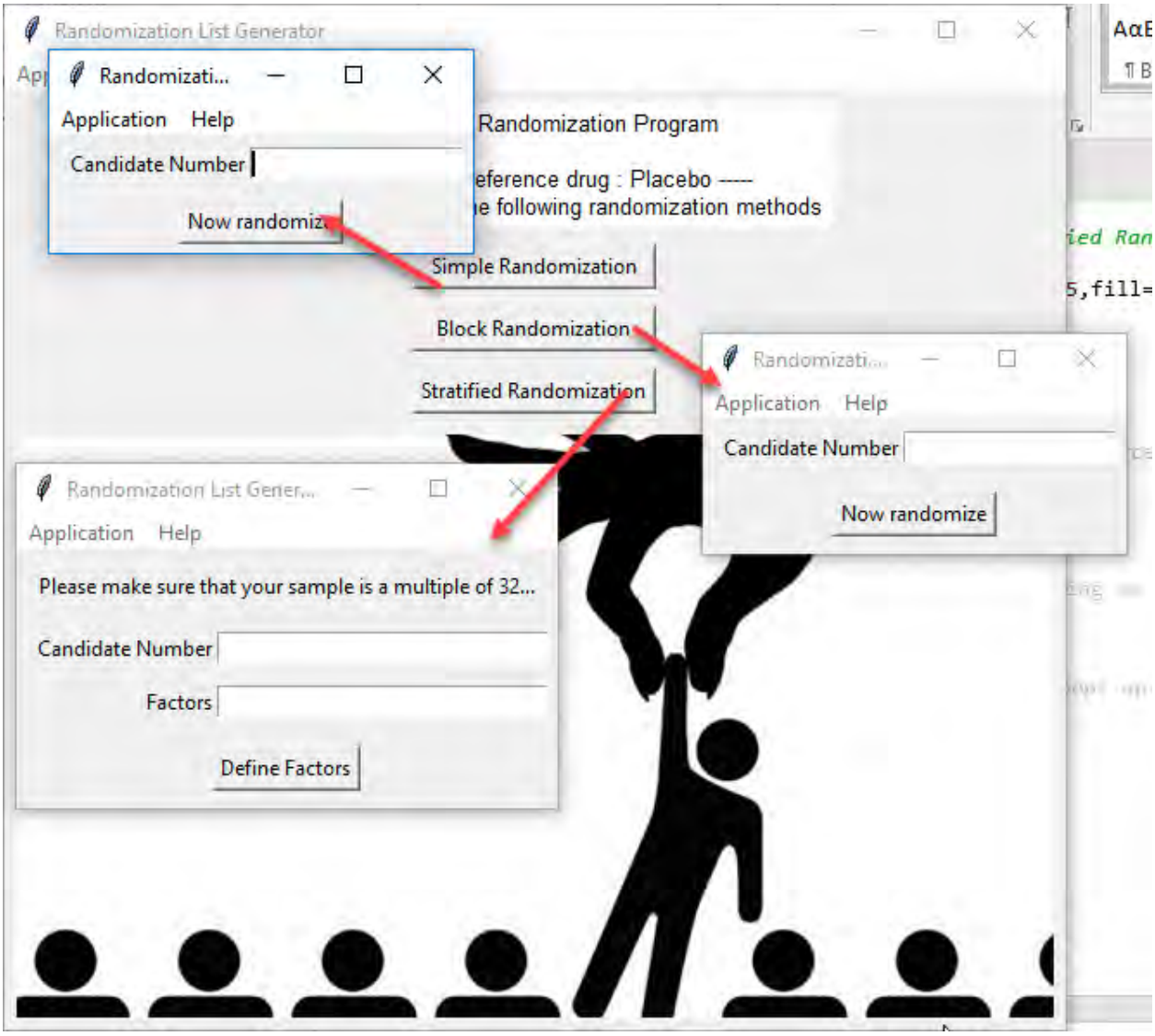
```
def intError(self): '''error:input should be multiple of 8'''

def intError1(self): '''error: input should be integer'''

def overflowError(self,strata): '''error:small sample size'''

def noFileError(self): '''no results created yet'''

def patternError(self): '''false input pattern for the factors during stratification'''
```

During the development, it was found that in order for the randomization tool to be effective and efficient, some essential requirements such as user interface and user experience should be implemented. Therefore, going back to Step 2 of the life cycle, I needed to analyze what is required to develop a graphical interface for the user.

Initialize GUI

Tkinter is Python's de-facto standard GUI (Graphical User Interface) package. It is a thin object-oriented layer on top of Tcl/Tk (python, 2017). Firstly, a management layout (create_Window) was designed by implementing three different buttons for each randomization method after creating a class RandomizeApp for this GUI (Tkinter, 2017) (Bernd Klein, 2018):

```
class RandomizeApp:

    def __init__(self, master):
        '''Initialize GUI window '''
        random.seed()

        frame = Frame(master)
        frame.pack()

        master.title('Randomization List Generator')

-----------------------------------

        #-----------Buttons Frame
        buttons = Frame(master)
        buttons.pack()

        self.b = Button(buttons, text="Simple Randomization",
command=self.create_windowA)
        self.b.pack( padx=5, pady=5,fill=X)

        self.s = Button(buttons, text="Block Randomization",
command=self.create_windowB)
        self.s.pack( padx=5, pady=5,fill=X)

        self.d = Button(buttons, text="Stratified Randomization",
command=self.create_windowC)
                    self.d.pack(padx=5, pady=5,fill=X)

--------------------------------------

            if __name__ == '__main__':

            #main call class RandomizeApp and creates an object

                root = Tk()

                bye = RandomizeApp(root)
                root.mainloop()#programm running on a loop until user clicks on exit
```




```
self.newWindow = Toplevel()#new window pops up in create_window command
```

Get User Input

Finally, for each of the three functions of randomization, makeForm(), getEnts() and fetch() is called to create forms to get information from the user:

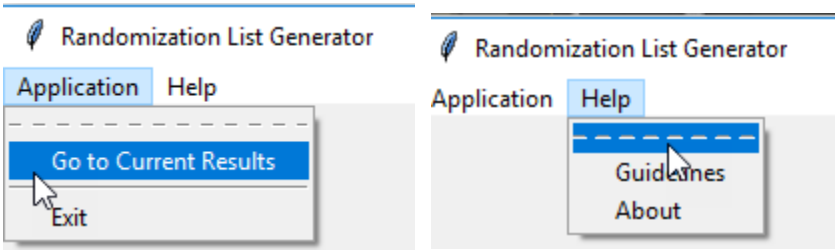
```
def makeform(self,master, fields):
''' Function that makes fields and entries for each method. Takes fields for
input '''

def getEnts(self):
''' Function calls fetch to create couple of fields and entries '''
```

```
def fetch(self,entries):
''' Function to get entries. Takes a list input from getEnts, returns a list
of entries '''
```

Display Menu

In order to have a better user interface, I implemented a menu that can be used for each separate window. The menu function includes the following:

```
def displayMenu(self,master): '''displays the menu'''
……..

def about(self): '''pops up info about project
 def goTo(self):      '''navigates to current results txt'''
def instructions(self): '''guidelines for the randomization techniques'''
```



Subsequently, each randomization was implemented with a writeFile function in order to save the results as .txt files at the users preferred folder:

```
def saveFile(self): ''' saveFile is called in every writeFile function '''
```

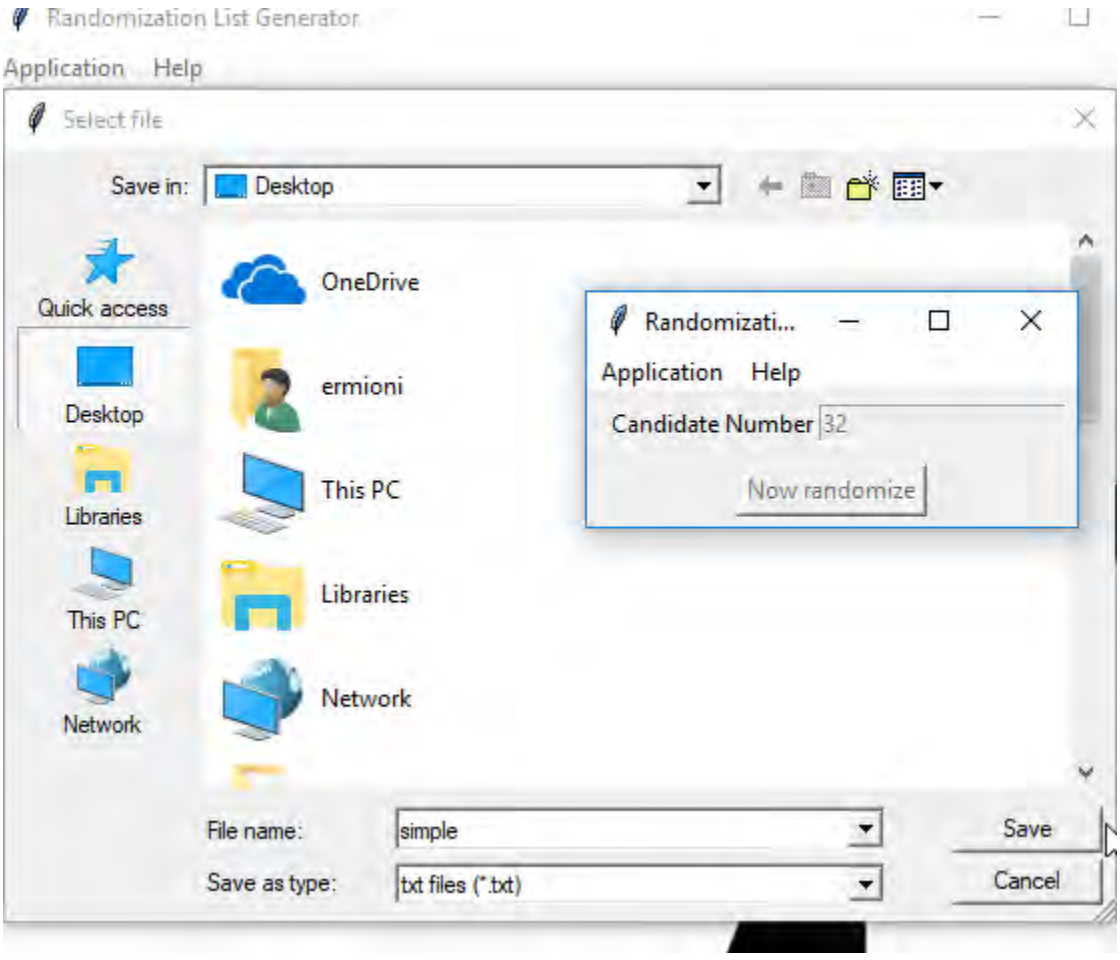
The Randomize App was tested successfully and is ready to be used (deployed) for any randomized trial with three treatment groups and a 3:3:2 ratio.

4. Results

The results of the RandomizeApp will be presented below for each for the three randomizations with indicative examples.

Manual for Simple Randomization:

In order to perform Simple Randomization, the user clicks on the designated button on the main screen of the programm. After that the user enters the number of candidates participating in the clinical trial which is multiple of 8 and clicks on the *'Randomize Now'* button. The Save as window opens up automatically and allows the user to choose the name of the .txt file and the folder destination:

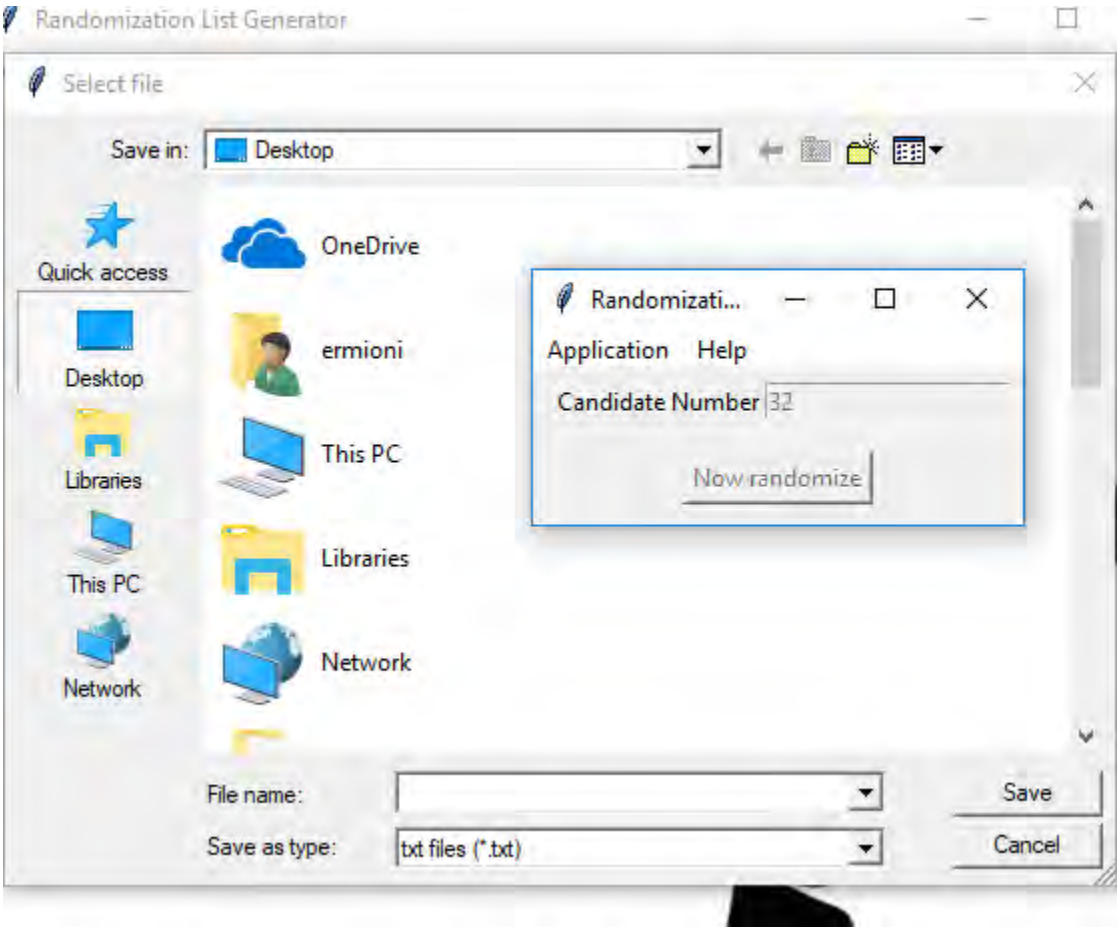


Finally, by clicking on the '*Go to Current Results'* on the menu bar or by navigating to the chosen folder the results are displayed as shown in the picture below:

```
Candidate 1 is assigned to the Placebo treatment.
Candidate 2 is assigned to the Placebo treatment.
Candidate 3 is assigned to the Test Drug treatment.
Candidate 4 is assigned to the Test Drug treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Reference Drug treatment.
Candidate 7 is assigned to the Test Drug treatment.
Candidate 8 is assigned to the Test Drug treatment.
Candidate 9 is assigned to the Reference Drug treatment.
Candidate 10 is assigned to the Reference Drug treatment.
Candidate 11 is assigned to the Test Drug treatment.
Candidate 12 is assigned to the Test Drug treatment.
Candidate 13 is assigned to the Reference Drug treatment.
Candidate 14 is assigned to the Placebo treatment.
Candidate 15 is assigned to the Test Drug treatment.
Candidate 16 is assigned to the Test Drug treatment.
Candidate 17 is assigned to the Test Drug treatment.
Candidate 18 is assigned to the Reference Drug treatment.
Candidate 19 is assigned to the Reference Drug treatment.
Candidate 20 is assigned to the Placebo treatment.
Candidate 21 is assigned to the Placebo treatment.
Candidate 22 is assigned to the Reference Drug treatment.
Candidate 23 is assigned to the Placebo treatment.
Candidate 24 is assigned to the Reference Drug treatment.
```

Manual for Block Randomization:

In order to perform Block Randomization, the user clicks on the designated button on the main screen of the programm. After that, the user enters the number of candidates participating in the clinical trial which is multiple of 8 and clicks on the *'Randomize Now'* button. The Save as window opens up automatically and allows the user to choose the name of the .txt file and the folder destination:



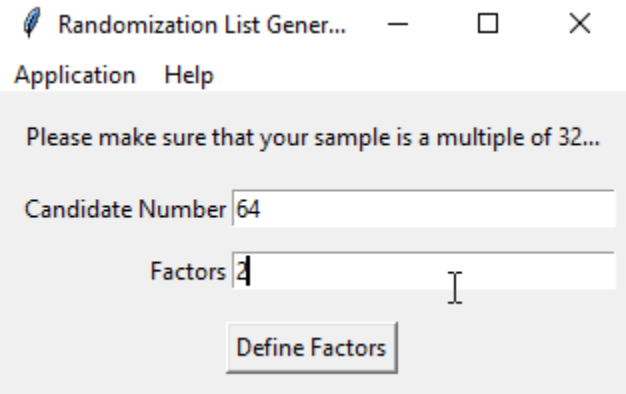
Finally, by clicking on the '*Go to Current Results'* on the menu bar or by navigating to the chosen folder the results are displayed as shown in the picture below:

```
Block 1:
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Test Drug treatment.
Candidate 3 is assigned to the Reference Drug treatment.
Candidate 4 is assigned to the Placebo treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Test Drug treatment.
Candidate 7 is assigned to the Reference Drug treatment.
Candidate 8 is assigned to the Placebo treatment.
Block 2:
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Placebo treatment.
Candidate 3 is assigned to the Placebo treatment.
Candidate 4 is assigned to the Reference Drug treatment.
Candidate 5 is assigned to the Test Drug treatment.
Candidate 6 is assigned to the Reference Drug treatment.
Candidate 7 is assigned to the Test Drug treatment.
Candidate 8 is assigned to the Reference Drug treatment.
Block 3:
Candidate 1 is assigned to the Reference Drug treatment.
Candidate 2 is assigned to the Reference Drug treatment.
Candidate 3 is assigned to the Test Drug treatment.
Candidate 4 is assigned to the Placebo treatment.
Candidate 5 is assigned to the Test Drug treatment.
Candidate 6 is assigned to the Placebo treatment.
Candidate 7 is assigned to the Test Drug treatment.
Candidate 8 is assigned to the Reference Drug treatment.
Block 4:
Candidate 1 is assigned to the Reference Drug treatment.
Candidate 2 is assigned to the Test Drug treatment.
Candidate 3 is assigned to the Test Drug treatment.
Candidate 4 is assigned to the Placebo treatment.
Candidate 5 is assigned to the Placebo treatment.
Candidate 6 is assigned to the Reference Drug treatment.
Candidate 7 is assigned to the Reference Drug treatment.
Candidate 8 is assigned to the Test Drug treatment.
```

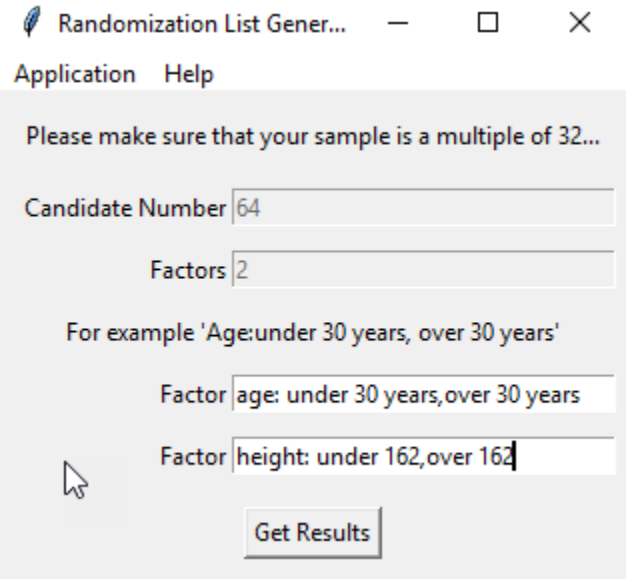
Manual for Stratified Randomization:

In order to perform Stratified Randomization, the user clicks on the designated button on the main screen of the programm. After that, the user enters the number of candidates participating in the clinical trial, which is multiple of 28, and enters how many factors/covariates are needed and clicks on the *'Get Factors'* button.

User enters the factors to create strata as follows:



The Save as window opens up automatically after clicking on *'Get Results'* and allows the user to choose the name of the .txt file and the folder destination:



The Stratification results display in the following form:

```
---------- Strata 1 :  under 30 years, under 162 ------
----------Factors: age: under 30 years,over 30 years - height: under 162,over 162
--- Block 1
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Test Drug treatment.
Candidate 3 is assigned to the Reference Drug treatment.
Candidate 4 is assigned to the Reference Drug treatment.
Candidate 5 is assigned to the Test Drug treatment.
Candidate 6 is assigned to the Placebo treatment.
Candidate 7 is assigned to the Reference Drug treatment.
Candidate 8 is assigned to the Placebo treatment.
--- Block 2
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Reference Drug treatment.
Candidate 3 is assigned to the Test Drug treatment.
Candidate 4 is assigned to the Reference Drug treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Placebo treatment.
Candidate 7 is assigned to the Test Drug treatment.
Candidate 8 is assigned to the Placebo treatment.
---------- Strata 2 :  under 30 years,over 162 ------
----------Factors: age: under 30 years,over 30 years - height: under 162,over 162
--- Block 1
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Placebo treatment.
Candidate 3 is assigned to the Test Drug treatment.
Candidate 4 is assigned to the Reference Drug treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Placebo treatment.
Candidate 7 is assigned to the Reference Drug treatment.
Candidate 8 is assigned to the Test Drug treatment.
--- Block 2
Candidate 1 is assigned to the Reference Drug treatment.
Candidate 2 is assigned to the Test Drug treatment.
Candidate 3 is assigned to the Placebo treatment.
Candidate 4 is assigned to the Test Drug treatment.
Candidate 5 is assigned to the Test Drug treatment.
Candidate 6 is assigned to the Reference Drug treatment.
Candidate 7 is assigned to the Reference Drug treatment.
Candidate 8 is assigned to the Placebo treatment.
---------- Strata 3 : over 30 years, under 162 ------
----------Factors: age: under 30 years,over 30 years - height: under 162,over 162
--- Block 1
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Placebo treatment.
Candidate 3 is assigned to the Test Drug treatment.
Candidate 4 is assigned to the Reference Drug treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Placebo treatment.
Candidate 7 is assigned to the Test Drug treatment.
Candidate 8 is assigned to the Reference Drug treatment.
--- Block 2
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Placebo treatment.
Candidate 3 is assigned to the Test Drug treatment.
Candidate 4 is assigned to the Reference Drug treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Reference Drug treatment.
Candidate 7 is assigned to the Test Drug treatment.
Candidate 8 is assigned to the Placebo treatment.
---------- Strata 4 : over 30 years,over 162 ------
----------Factors: age: under 30 years,over 30 years - height: under 162,over 162
--- Block 1
Candidate 1 is assigned to the Reference Drug treatment.
Candidate 2 is assigned to the Placebo treatment.
Candidate 3 is assigned to the Reference Drug treatment.
Candidate 4 is assigned to the Test Drug treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Test Drug treatment.
Candidate 7 is assigned to the Test Drug treatment.
Candidate 8 is assigned to the Placebo treatment.
--- Block 2
Candidate 1 is assigned to the Test Drug treatment.
Candidate 2 is assigned to the Test Drug treatment.
Candidate 3 is assigned to the Reference Drug treatment.
Candidate 4 is assigned to the Placebo treatment.
Candidate 5 is assigned to the Reference Drug treatment.
Candidate 6 is assigned to the Test Drug treatment.
Candidate 7 is assigned to the Reference Drug treatment.
Candidate 8 is assigned to the Placebo treatment.
```

5. Conclusions

Clinical trials are performed to determine important pathophysiological effects of new drugs or new therapies between different treatment groups. It is vital to have an efficient tool to perform a randomization task in the beginning of every study.

The Randomization App presented in this work is a tool that can be used during the selection process of a clinical trial. It provides protection from selection and confounding biases by blinding the process of assigning the candidates to the three treatment groups.

To summarize, this software offers three choices to the user, for three different randomization techniques (Simple, Block, Stratified) and provides the results fast and efficiently, taking into consideration all the limitations to perform a 3:3:2 randomization.

Ultimately, this tool has a lot of functional features; however there is room for improvement. An additional function that could be implemented is to provide the choice of desirable distribution ratios and different therapeutic groups to the user.

Bibliography

1. Altman, D. G., 1991. 5.7.2 Block (or restricted) randomization. In: *Practical Statistics for Medical Research.* s.l.:Chapman & Hall/CRC, pp. 87-89.

2. Bernd Klein, B., 2018. *Python Tkinter Course.* [Online]
Available at: https://www.python-course.eu/python_tkinter.php

3. Day SJ, A. D., 2000. Blinding in clinical trials and other studies. *BMJ : British Medical Journal,* 7259(504), p. 321.

4. eclipse.org, 2018. *Eclipse IDE.* s.l.

5. Hussung, T., 2016. *What Is the Software Development Life Cycle?.* [Online]
Available at: https://online.husson.edu/software-development-cycle/

6. Kowald, A., 2017. Introduction to Computer Programming in Python . In: s.l.:s.n.

7. Novartis, 2017. *www.mednet.gr.* [Online]
Available at:
http://www.mednet.gr/application/modules/Wow/externals/files/Dikaioma_Clinical%20Trials_(GR1707667598).pdf

8. Pennsylvania State University, n.d. [Online]
Available at: https://onlinecourses.science.psu.edu/stat507/node/34/

9. Python.org, 2008. *Python.* s.l. Patent No. 3.0.

10. python, 2017. *Python tkinter.* [Online]
Available at: https://wiki.python.org/moin/TkInter

11. Suresh, K., 2011. An overview of randomization techniques: An unbiased assessment of outcome in clinical research. *J Human Reproductive Sciences,* p. April.

12. Tkinter, 2017. *tutoriaspoint.* [Online]
Available at: https://www.tutorialspoint.com/python/python_gui_programming.htm

13. wikipedia, n.d. *wikipedia.org.* [Online]
Available at: https://en.wikipedia.org/wiki/Selection_bias

Appendix: Code

```
from tkinter import *
from tkinter import messagebox
from tkinter import filedialog
import os.path
import errno
import random
import itertools
import re
import webbrowser


class RandomizeApp:

    def __init__(self, master):
        '''Initialize GUI window '''
        random.seed()

        frame = Frame(master)
        frame.pack()

        master.title('Randomization List Generator')

        #------------Menu
        self.displayMenu(master)

        #-----------Welcome Message Frame
        welcome = Frame(master)
        welcome.pack(side= TOP)

        self.message = ''' Welcome to the 3:3:2 Randomization Program\n
        ----- Test drug : Reference drug : Placebo -----\nPlease select one of the
following randomization methods'''
        self.msg = Message(welcome, text = self.message)
        self.msg.config(bg='white', font=('arial', 10),width = 450)
        self.msg.pack(side=TOP,padx=5,pady=5,fill=X)

        #-----------Buttons Frame
        buttons = Frame(master)
        buttons.pack()

        self.b = Button(buttons, text="Simple Randomization",
command=self.create_windowA)
        self.b.pack( padx=5, pady=5,fill=X)

        self.s = Button(buttons, text="Block Randomization",
command=self.create_windowB)
        self.s.pack( padx=5, pady=5,fill=X)

        self.d = Button(buttons, text="Stratified Randomization",
command=self.create_windowC)
```

```
        self.d.pack(padx=5, pady=5,fill=X)

        #--------Photo
        photo_frame = Frame(master)
        photo_frame.pack()

        self.photo = PhotoImage(file="random.png")
        self.ph = Label(photo_frame, image=self.photo)
        self.ph.pack(padx=5 ,pady=5)

    def displayMenu(self,master):
        ''' Defines MENU for frames'''
        self.menu = Menu(master)
        master.config(menu = self.menu)

        appMenu = Menu(self.menu)
        self.menu.add_cascade(label='Application', menu=appMenu)
        appMenu.add_command(label='Go to Current Results',command=self.goTo)
        appMenu.add_separator()
        appMenu.add_command(label='Exit', command=master.quit)

        helpMenu = Menu(self.menu)
        self.menu.add_cascade(label='Help', menu = helpMenu)
        helpMenu.add_command(label='Guidelines',command=self.instructions)
        helpMenu.add_command(label='About', command= self.about)



    def about(self):
        messagebox.showinfo("Title", "Project by Ermioni Kritsinioti\nMSc Biomath ")
    def goTo(self):


        try:
            if os.path.exists(self.folder_path):
                webbrowser.open(self.folder_path)

        except AttributeError:
            self.noFileError()

    def instructions(self):
        ''' Instructions for the three randomization methods  '''
        self.newWindow = Toplevel()


        self.message = '''Randomization as a method of experimental control has been
extensively used in human clinical trials and other biological experiments.
        It prevents the selection bias and insures against the accidental bias.
        It produces the comparable groups and eliminates the source of bias in
treatment assignments

        Simple Randomization: Randomization based on a single sequence of random
assignments. In order to maintain a 3:3:2 ratio for Raference Drug:Test Drug:Placebo
, users are only allowed to enter candidate numbers that are multiple of 8 and
preferably have a sample size that is above 24.
```

```
        Block Randomization: The block randomization method is designed to randomize
subjects into groups that result in equal sample sizes. This method is used to ensure
a balance in sample size across groups over time. Block size is 8 and ratio 3:3:2 is
kept between all blocks.

        Stratified Randomization: The stratified randomization method addresses the
need to control and balance the influence of covariates. This method can be used to
achieve balance among groups in terms of subjects baseline characteristics
(covariates). Specific covariates must be identified by the researcher who
understands the potential influence each covariate has on the dependent variable.
        Stratified randomization is achieved by generating a separate block for each
combination of covariates or strata, and subjects are assigned to the appropriate
block of covariates. After all subjects have been identified and assigned into
blocks, simple randomization is performed within each block to assign subjects to one
of the groups.
        Note that in order to perform stratification , user needs to enter a
candidate number that is a multiple of 28. Factors should not be more than 4-5 in
order to have coverage for all stratas. For example if users need 2
covariates/factors like Age:under 30 years,over 30 years and Weight:Under 60 kgs,over
60kgs, our stratas will equal to 2x2 = 4 sratifications.
        Therefore Strata 1 is under 30 years/under 60kgs, strata 2 is under 30
years/over 60kgs, strata 3 is over 30 years/over 60 kgs and strata 4 is over 30 years
/over 60kgs. '''
        self.instructs = Message(self.newWindow, text = self.message)
        self.instructs.config(bg='lightgrey', font=('arial', 10),width = 450)
        self.instructs.pack(side=TOP,padx=5,pady=5,fill=X)
    def fetch(self,entries):
        ''' Function to get entries.Takes a list input from getEnts '''
        self.texts = []
        for entry in entries:
            field = entry[0]

            text  = entry[1].get()

            self.texts.append(text)
            print('%s: "%s"' % (field, text))
            entry[1].config(state = 'disabled')
        return self.texts

    def makeform(self,master, fields):
        ''' Function that makes fields and entries for each method. Takes fields for
input '''
        self.entries = []


        for field in fields:
            row = Frame(master)
            lab = Label(row, width=15, text=field, anchor='e')
            ent = Entry(row)
            ent.focus()
            row.pack(side=TOP, fill=X, padx=5, pady=5)
            lab.pack(side=LEFT)
            ent.pack(side=RIGHT, expand=YES, fill=X)
            self.entries.append((field, ent))
```

```
        return self.entries

    def getEnts(self):

        self.entries_list = self.fetch(self.ents)
        return self.entries_list
    def saveFile(self):

        self.filename =  filedialog.asksaveasfilename(initialdir = "/",title =
"Select file",filetypes = (("txt files","*.txt"),("all files","*.*")))
        return self.filename
    def showStatus(self):

        self.newWindow.lift()
        status = Label(self.newWindow, text='Results saved in
{}'.format(self.folder_path),bd=1,relief=SUNKEN,anchor = W)
        status.pack(side=BOTTOM, fill=X)
        status.after(3000, lambda: status.destroy())#close status after 3secs
        return None
##################################  Simple  ######################
    def Simple(self,n):
        #treatments = ['test_drug' ,'ref_drug','placebo']
        treatA = int(round(n*3./8)) # separating the size of each group (ratios
3:2:2)
        treatB = int(round(n*3./8))
        treatC = int(round(n*2./8))
        print('Test Drug: {} Reference Drug: {} Placebo: {}
\n'.format(treatA,treatB,treatC))

        self.population = ['Test Drug']*treatA + ['Reference Drug']*treatB +
['Placebo']*treatC
        random.shuffle(self.population)
        #print(self.population)

        return self.population

    def writeFileSimple(self,results):
        '''Takes a list as input and creates a file and a folder if it does not exist
'''
        #homepath = os.getenv('USERPROFILE')
        #folder_path =
os.path.join(homepath+'\Desktop\Randomize_App\\resultSimple.txt')

        self.folder_path = self.saveFile()+'.txt'
        print(self.folder_path)

        try:
            if not os.path.exists(self.folder_path):
                with open(self.folder_path,'w') as wf:

                                    for index,item in enumerate(results, start=1):
                                        items = 'Candidate {} is assigned to the {}
treatment.'.format(index,item)
                                        wf.write("%s\n" % items)
```

```
                wf.close()
                self.showStatus()
            else:
                messagebox.showerror('Error', 'File name already exists!')
                self.newWindow.lift()
        except OSError as exc:
            if exc.errno != errno.EEXIST:
                    raise


    def simpleRand(self):
        self.b1.configure(state = DISABLED)#disables button
        patients = int(self.getEnts()[0])

        if patients<8 or patients%8!=0:
            self.intError()
        else:
            s_list = self.Simple(patients)
            self.writeFileSimple(s_list)

        return None

    def create_windowA(self):
        ''' Creates window and fields for the simple randomization '''
        print('A clicked')
        fields = ['Candidate Number']
        self.newWindow = Toplevel()
        self.displayMenu(self.newWindow)

        self.ents = self.makeform(self.newWindow, fields)
        self.newWindow.bind('<Return>', self.getEnts)
        self.b1 = Button(self.newWindow,text = 'Now randomize',command=
self.simpleRand)
        self.b1.pack(side=TOP, padx=5, pady=5)


################################Blocked  ##########################################
    def writeFileBlock(self,results):
        '''takes a 2 - dimensional list and writes a file for Block randomization '''
        self.folder_path = self.saveFile()+'.txt'
        print(self.folder_path)

        try:
            if not os.path.exists(self.folder_path):
                with open(self.folder_path,'w') as wf:

                            for index,item in enumerate(results, start=1):
                                items = 'Block {}:'.format(index)
                                wf.write("%s\n" % items)
                                for i,drug in enumerate(item,start=1):
                                    result = 'Candidate {} is assigned to the {}
treatment.'.format(i,drug)
                                    wf.write("%s\n" % result)
                wf.close()
                self.showStatus()
```

```
            else:
                messagebox.showerror('Error', 'File name already exists!')
                self.newWindow.lift()
        except OSError as exc:
            if exc.errno != errno.EEXIST:
                    raise

    def Blocked(self,n):
        blocks_list = []
        blocks = 1
        temp_block_size = 8
        while n:


            if temp_block_size>n:
                temp_block_size = n
                #print("Corrected")

            n = n - temp_block_size
            print("----- Block {} with block size {} -----
\n".format(blocks,temp_block_size))
            x_block = self.Simple(temp_block_size)
            blocks_list.append(x_block)
            blocks +=1

        #print(blocks_list)
        return blocks_list

    def blockRand(self):
        self.b1.configure(state = DISABLED)
        patients = int(self.getEnts()[0])

        if patients<8 or patients%8!=0:
            self.intError()
        else:
            b_list = self.Blocked(patients)
            self.writeFileBlock(b_list)

        return None

    def create_windowB(self):
        print('B clicked')
        fields = ['Candidate Number']
        self.newWindow = Toplevel()#new window pops up
        self.displayMenu(self.newWindow)

        self.ents = self.makeform(self.newWindow, fields)
        self.newWindow.bind('<Return>', self.getEnts)
        self.b1 = Button(self.newWindow,text = 'Now randomize', command =
self.blockRand)
        self.b1.pack(side=TOP, padx=10, pady=10)
#####################################Sratification
#################################
    def writeFileStrata(self,results,factors,levels):
```

```
        #results is a 3d list of stratas and blocks , factors is a string ,levels is
a list of the paired stratas
        self.folder_path = self.saveFile()+'.txt'
        print(self.folder_path)

        try:
            if not os.path.exists(self.folder_path):
                with open(self.folder_path,'w') as wf:
                    for index,item in enumerate(results, start=1):
                        current_level = ",".join(levels[index-1])
                        items = '----------- Strata {} : {} ------\n-----------
Factors: {}'.format(index,current_level,factors)
                        wf.write("%s\n" % items)
                        for i,drugs in enumerate(item,start=1):
                            cur_block = '--- Block {}'.format(i)
                            wf.write("%s\n" % cur_block)
                            for j,drug in enumerate(drugs,start=1):
                                result = 'Candidate {} is assigned to the {}
treatment.'.format(j,drug)
                                wf.write("%s\n" % result)
                wf.close()
                self.showStatus()
            else:
                messagebox.showerror('Error', 'File name already exists!')
                self.newWindow.lift()
        except OSError as exc:
            if exc.errno != errno.EEXIST:
                    raise

    def strataRand(self):
        n = self.patients
        self.b2.configure(state = DISABLED)
        #self.b2.destroy()
        levels = []

        strata = 1
        factors = self.getEnts()
        for factor in factors:
            pattern = re.compile(r'(.+):(.+),(.+)')
            matches = pattern.search(factor)
            if not matches:
                self.patternError()

        for i in range(len(factors)):
            level = factors[i].split(':')[1].split(',')
            levels.append(level)
            strata *= len(level)

        levels = list(itertools.product(*levels))
        print('Levels is {} and strata is {}'.format(levels,strata))

        try:
            strata_size = int(n/strata) #separate candidates into stratas
            assert(strata_size >= 8),'Small sample size!'
            factors = ' - '.join(factors)
```

```
            strata_results = []
            i=0
            while i!=strata:

                print("Factors: ",factors)
                ##For every strata we make a blocked randomization with random sized
blocks
                current_level = ",".join(levels[i])
                print("Strata {}: {}".format(i+1,current_level))
                print("================================")
                res = self.Blocked(strata_size)
                strata_results.append(res)
                i+=1
            print(strata_results)


            if i==strata:
                self.writeFileStrata(strata_results,factors,levels)

        except AssertionError as a:
            print('Too small sample size for {} stratifications !'.format(strata),a)
            self.overflowError(strata)

    def getFactors(self):
        ''' get factors for stratification '''
        self.b1.destroy()
        try:
            self.patients = int(self.getEnts()[0])
            factors = int(self.getEnts()[1])
            assert(self.patients>8 and self.patients%32==0),'Multiple of 32 please!'

            msg = Label(self.newWindow,text="For example 'Age:under 30 years, over 30
years'")
            msg.pack(padx=5,pady=5)
            fields = ["Factor"]*factors

            self.ents = self.makeform(self.newWindow, fields)
            self.newWindow.bind('<Return>', self.getEnts)
            self.b2 = Button(self.newWindow,text='Get Results',command=
self.strataRand)
            self.b2.pack(padx=10, pady=10)

        except AssertionError as e:
            print(e)
            self.intError()

        except ValueError:
            self.intError1()

        return None

    def create_windowC(self):

        print('C clicked')
        fields = ['Candidate Number','Factors']
```

```
        self.newWindow = Toplevel()
        self.displayMenu(self.newWindow)

        strata_message = Label(self.newWindow, text = 'Please make sure that your
sample is a multiple of 32...')
        strata_message.pack(side=TOP, padx=10, pady=10)

        self.ents = self.makeform(self.newWindow, fields)
        self.newWindow.bind('<Return>', self.getEnts)
        self.b1 = Button(self.newWindow,text = 'Define Factors', command =
self.getFactors)
        self.b1.pack(side=TOP, padx=10, pady=10)
########################################## Errors

    def intError(self):
        messagebox.showerror("Error", "Candidate number should be multiple of 8!")
        self.newWindow.destroy()
    def intError1(self):
        messagebox.showerror("Error", "Input should be integers only!")
        self.newWindow.destroy()
    def overflowError(self,strata):
        messagebox.showerror("Error", "Too small sample size for {} stratifications !
Small sample size!".format(strata))
        self.newWindow.destroy()
    def noFileError(self):
        messagebox.showerror('Error', 'No such file or directory created')
    def patternError(self):
        messagebox.showerror("Error", "Wrong input pattern for the factors!\nPattern
should match following: 'Age:under 30 years,over 30 years' etc.")
        self.newWindow.destroy()
#############################################################################################
######

if __name__ == '__main__':
    root = Tk()
    bye = RandomizeApp(root)


    root.mainloop()
```