

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών:
Graphical Weighting Method, Delphi Method, Fixed Point Scoring
και Swing Weighting Method σε περιβάλλον Python

Εργασία 13

Μπιρμπουτσάκης Χρήστος – Κωνσταντίνος (2017010036)

Σκουλάξινος Παναγιώτης – Χαράλαμπος (2017010035)

Περιεχόμενα

Εισαγωγή	
Απαιτούμενες βιβλιοθήκες.....	3
Fixed Point Scoring	
Η Μέθοδος.....	4
Παράδειγμα.....	4
Υλοποίηση της μεθόδου σε Python.....	5
Περιγραφή.....	5
Εκτέλεση προγράμματος.....	7
Swing Weighting Method	
Η Μέθοδος.....	8
Παράδειγμα.....	8
Υλοποίηση της μεθόδου σε Python.....	10
Περιγραφή.....	10
Εκτέλεση προγράμματος.....	14
Graphical Weighting Method	
Η Μέθοδος.....	16
Παράδειγμα.....	16
Υλοποίηση της μεθόδου σε Python.....	17
Περιγραφή.....	17
Εκτέλεση Προγράμματος.....	19
Delphi Weighting Method	
Η Μέθοδος.....	21
Παράδειγμα.....	21
Υλοποίηση της μεθόδου σε Python.....	23
Περιγραφή.....	23
Εκτέλεση προγράμματος.....	27
Βιβλιογραφία	

Εισαγωγή

Στη συγκεκριμένη εργασία παρουσιάζονται 4 μέθοδοι απόδοσης βαρών, οι Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method. Συγκεκριμένα περιγράφεται ο τρόπος λειτουργίας και εφαρμογής της κάθε μεθόδου σε ξεχωριστό κεφάλαιο, όπως επίσης και η υλοποίηση αυτών σε περιβάλλον Python. Η κάθε μέθοδος παρουσιάζεται σε ξεχωριστό κεφάλαιο.

Απαιτούμενες βιβλιοθήκες

Για να δουλέψουν τα προγράμματα σωστά, χρειάζεται να υπάρχουν στο installation της Python 3.8.2 οι εξής βιβλιοθήκες:

- pandas 1.0.5
- xlrd 1.2.0
- numpy 1.17.4
- matplotlib 3.2.2

Τα προγράμματα έχουν δοκιμαστεί και τρέχουν στις παραπάνω εκδόσεις των βιβλιοθηκών. Είναι πιθανό ότι θα τρέχουν και σε άλλες εκδόσεις, αλλά το γεγονός αυτό δεν έχει δοκιμαστεί.

Fixed Point Scoring

Η Μέθοδος

Στη μέθοδο απόδοσης βαρών Fixed Point Scoring ο αποφασίζων καλείται να μοιράσει ένα πεπερασμένο πλήθος πόντων (συνήθως 100 αλλά δεν είναι δεσμευτικό) στις διάφορες εναλλακτικές περιπτώσεις ή κριτήρια τα οποία καλείται να εξάγει βάρη. Παρακάτω βλέπουμε ένα παράδειγμα της μεθόδου απόδοσης βαρών Fixed Point Scoring.

Παράδειγμα

Πίνακας 1: Οι εναλλακτικές του προβλήματος και οι πόντοι που τους ορίζουμε

Εναλλακτικές	Πόντοι
Εναλλακτική 1	40
Εναλλακτική 2	110
Εναλλακτική 3	35
Εναλλακτική 4	15

Στο συγκεκριμένο παράδειγμα, όπως φαίνεται στον πίνακα 1, μοιράζουμε 200 πόντους σε 4 εναλλακτικές. Για να εξάγουμε βάρη, απλά κανονικοποιούμε τους παραπάνω αριθμούς, ώστε να αθροίζουν στη μονάδα.

Πίνακας 2: Βάρη. Τα αποτελέσματα της μεθόδου Fixed Point Scoring

Εναλλακτικές	Βάρη
Εναλλακτική 1	$40/200 = 0.2$ ή 20%
Εναλλακτική 2	$110/200 = 0.55$ ή 55%
Εναλλακτική 3	$35/200 = 0.175$ ή 17.5%
Εναλλακτική 4	$15/200 = 0.075$ ή 7.5%

Υλοποίηση της μεθόδου σε Python

Περιγραφή

Αρχικά εισάγουμε δεδομένα στο πρόγραμμα μέσω ενός υπολογιστικού φύλλου Excel. Η μορφή του αρχείου Excel φαίνεται στο στιγμιότυπο 1. Έπειτα, χρησιμοποιώντας τη βιβλιοθήκη *pandas* δημιουργούμε ένα πλαίσιο δεδομένων (data frame) με βάσει τα δεδομένα του Excel ώστε να μπορούμε να τα επεξεργαστούμε (στιγμιότυπο 2).

	A	B
1	alternatives	points
2	Alt1	40
3	Alt2	110
4	Alt3	35
5	Alt4	15

Στιγμιότυπο 1: Μορφή του αρχείου που διαβάζει το πρόγραμμα. Στην αριστερή στήλη είναι τα ονόματα των εναλλακτικών και στη δεξιά στήλη οι πόντοι που τους κατανέμουμε.

```
# Import data
Data = pd.read_excel(r'fixedPointAlts.xlsx', index_col=0)

# Convert point column to list, save alt names to list
Points = Data['points'].tolist()
Alts = Data.index.tolist()
```

Στιγμιότυπο 2: Διαβάζουμε το αρχείο και το αποθηκεύουμε στη μεταβλητή *Data*. Έπειτα αποθηκεύουμε τους πόντους στη λίστα *Points* και τα ονόματα των εναλλακτικών στη λίστα *Alts*.

Έπειτα, υπολογίζουμε το πλήθος των πόντων που ο χρήστης επέλεξε να κατανέμει, και τους αποθηκεύει στη μεταβλητή *total*. Μετά κανονικοποιούμε τους πόντους και τους αποθηκεύουμε στη λίστα *Weight*.

```
# Calculate total points
total = 0
for point in Points:
    total += point

# Normalize points
Weight = []
for point in Points:
    Weight.append(point / total)
```

Στιγμιότυπο 3: Υπολογισμός πλήθους πόντων και κανονικοποίηση αυτών

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών: Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method σε περιβάλλον Python

Τέλος, το πρόγραμμα δημιουργεί γράφημα “plot_ημερομηνία_ώρα.png”, τυπώνει τα βάρη και τα εξάγει σε ένα αρχείο Excel με όνομα “output_ημερομηνία_ώρα.xlsx”.

```
# Plot results
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.barh(Alts,Weight)
ax.set_ylabel('Alternatives')
ax.set_xlabel('Weights')

# Get date+time for filename
dateTimeObj = datetime.now()
timestampStr = dateTimeObj.strftime("%d-%m-%Y_%H:%M:%S")

# Save to png
plt.savefig('plot_' + timestampStr + '.png',bbox_inches='tight')

# Save weights to Excel && print results
print(pd.DataFrame(Weight,Alts))
pd.DataFrame(Weight,Alts).to_excel("output_" + timestampStr + '.xlsx', header=False)
|
```

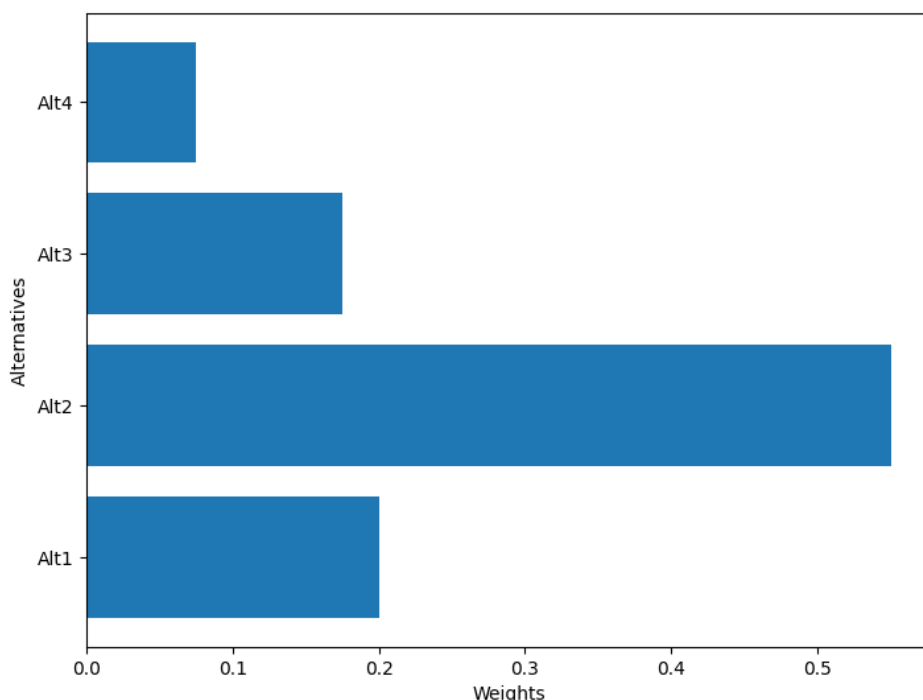
Στιγμιότυπο 4: Εκτύπωση, δημιουργία γραφήματος και εξαγωγή των αποτελεσμάτων σε αρχείο

Εκτέλεση προγράμματος

Όπως αναφέραμε παραπάνω, το πρόγραμμα αρχικά διαβάζει και εισάγει δεδομένα από το αρχείο *fixedPointAlts.xlsx* το οποίο πρέπει να βρίσκεται στον ίδιο φάκελο με το εκτελέσιμο αρχείο. Το πρόγραμμα εκτελεί την διαδικασία που περιγράφεται παραπάνω, τυπώνει στην οθόνη τα ονόματα των εναλλακτικών με τα βάρη που προέκυψαν δίπλα τους (στιγμιότυπο 5) και τα εξάγει σε γράφημα και σε αρχείο Excel (στιγμιότυπα 6 και 7).

```
panos@panos-pc: ~/Dev/PyCharmProjects/legendary-rotary-phone/Fixed Point Scoring$ python3
fixedPointScoring.py
Gtk-Message: 16:06:40.509: Failed to load module "appmenu-gtk-module"
0
Alt1  0.200
Alt2  0.550
Alt3  0.175
Alt4  0.075
/home/panos/.local/lib/python3.8/site-packages/matplotlib/backends/backend_gtk3.py:195: W
arning: Source ID 7 was not found when attempting to remove it
  Glib.source_remove(self._idle_draw_id)
panos@panos-pc: ~/Dev/PyCharmProjects/legendary-rotary-phone/Fixed Point Scoring$
```

Στιγμιότυπο 5: Η έξοδος του προγράμματος. Τα δεδομένα που εισάγουμε είναι όμοια με τη θεωρητική επίλυση του προβλήματος.



Στιγμιότυπο 6: Το γράφημα που εξάγει το πρόγραμμα

	A	B
1	Alt1	0.2
2	Alt2	0.55
3	Alt3	0.175
4	Alt4	0.075

Στιγμιότυπο 7: Το αρχείο Excel που εξάγει το πρόγραμμα

Swing Weighting Method

Η Μέθοδος

Στη μέθοδο απόδοσης βαρών swing ο αποφασίζων έχει το δικαίωμα να μεγιστοποιήσει μόνο ένα από τα διαθέσιμα κριτήρια, θεωρώντας ότι τα υπόλοιπα κριτήρια είναι στην χειρότερη μορφή που τα συναντάμε. Έπειτα, αποφασίζων καλείται να αποφασίσει πόσο τον ενδιαφέρει η βελτίωση των υπολοίπων κριτηρίων σε σχέση με το κριτήριο που επέλεξε ως σημαντικότερο. Παρακάτω βλέπουμε ένα παράδειγμα της μεθόδου απόδοσης βαρών swing.

Παράδειγμα

Πίνακας 3: Παράδειγμα εφαρμογής μεθόδου απόδοσης βαρών Swing

Εναλλακτικές	Κόστος (ευρώ)	Αξιοπιστία	Χρόνος (ώρες)
A	2200	Καλή	60
B	2500	Εξαιρετική	25
Γ	3000	Αποδεκτή	40

Ξεκινάμε φτιάχνοντας μια εικονική εναλλακτική που έχει τις χειρότερες τιμές των κριτηρίων και την ονομάζουμε benchmark.

Πίνακας 4: Πίνακας 3 με επιπλέον εναλλακτική benchmark

Εναλλακτικές	Κόστος (ευρώ)	Αξιοπιστία	Χρόνος (ώρες)
(Benchmark)	3000	Αποδεκτή	60
A	2200	Καλή	60
B	2500	Εξαιρετική	25
Γ	3000	Αποδεκτή	40

Έπειτα φτιάχνουμε τον πίνακα της συγκεκριμένης μεθόδου, ο οποίος για κάθε εναλλακτική έχει το ένα βέλτιστο και τα υπόλοιπα δύο τα χειρότερα. Έστω ότι ο αποφασίζων θεωρεί το κόστος βασικότερο κριτήριο, την αξιοπιστία 40% σημαντική σε σχέση με το κόστος και το χρόνο 10%. Παρακάτω βλέπουμε τον πίνακα της swing που προκύπτει από τις προτιμήσεις του αποφασίζοντος.

Πίνακας 5: Πίνακας μεθόδου Swing

Εναλλακτικές	Κόστος (ευρώ)	Αξιοπιστία	Χρόνος (ώρες)	Κατάταξη
(Benchmark)	3000	Αποδεκτή	60	0
Κόστος	2200	Αποδεκτή	60	100
Αξιοπιστία	3000	Εξαιρετική	60	40
Χρόνος	3000	Αποδεκτή	25	10

Με βάσει, λοιπόν, την στήλη “Κατάταξη” προκύπτουν και τα τελικά βάρη και τερματίζει η μέθοδος απόδοσης βαρών Swing. Τα βάρη προκύπτουν διαιρώντας το εκάστοτε νούμερο - ποσοστό της κατάταξης με το άθροισμα των ποσοστών της στήλης αυτής.

Πίνακας 6: Βάρη. Το αποτέλεσμα της μεθόδου Swing

Κριτήρια	Βάρη
Κόστος	$100/150 = 0.666$ ή 66.6%
Αξιοπιστία	$40/150 = 0.267$ ή 26.7%
Χρόνος	$10/150 = 0.067$ ή 6.7%

Υλοποίηση της μεθόδου σε Python

Περιγραφή

Αρχικά εισάγουμε δεδομένα στο πρόγραμμα μέσω ενός υπολογιστικού φύλλου Excel. Η μορφή του αρχείου Excel φαίνεται στο στιγμιότυπο 8. Έπειτα, χρησιμοποιώντας τη βιβλιοθήκη *pandas* δημιουργούμε ένα πλαίσιο δεδομένων (data frame) με βάσει τα δεδομένα του Excel ώστε να μπορούμε να τα επεξεργαστούμε (στιγμιότυπο 9).

	A	B	C	D
1		Cost	Reliability	Time
2	A	2200	1	60
3	B	2500	0	25
4	C	3000	2	40

Στιγμιότυπο 8. Μορφή του αρχείου που διαβάζει το πρόγραμμα. Στην πρώτη στήλη ξεκινώντας από τη δεύτερη γραμμή είναι τα ονόματα των εναλλακτικών και στην πρώτη γραμμή ξεκινώντας από τη δεύτερη στήλη είναι τα ονόματα των κριτηρίων

```
# Import file "swingAlts.xlsx"
Data = pd.read_excel(r'swingAlts.xlsx', index_col=0)
print('Imported file swingAlts.xlsx!')
print(Data)
r, c = Data.shape
```

Στιγμιότυπο 9: Διαβάζουμε το αρχείο και το αποθηκεύουμε στη μεταβλητή *Data*. Την τυπώνουμε και αποθηκεύουμε τις διαστάσεις του πίνακα στις μεταβλητές *r, c*.

Έπειτα, αποθηκεύουμε τα ονόματα των εναλλακτικών και των κριτηρίων στις λίστες *Alts* και *Criteria*, αρχικοποιούμε τις λίστες *benchmark* και *Swings* και γεμίζουμε την πρώτη με τα μέγιστα και τον τελευταίο με τα ελάχιστα του *Data*. Γεμίζουμε μια νέα λίστα *Matrix* με ό,τι έχει η *benchmark* και μετά αντικαθιστούμε στη διαγώνιο της *Matrix* τα καλύτερα από τη λίστα *Swings*. Έτσι, η *Matrix* προκύπτει να είναι ο ίδιος πίνακας με τον πίνακα 5 στη θεωρητική επίλυση. Δημιουργούμε νέο data frame με τη λίστα *Matrix*, το ονομάζουμε *SwingData* και το τυπώνουμε.

```
# Determine most valued criterion
try:
    choice = int(input("Please pick the criteria that you value most (Enter number 1,2,...): ")) - 1
except ValueError:
    print("Error 100: NaN")
    sys.exit(100)
print("You value " + Criteria[choice] + " most.\n")

# Determine the value of swing from one criterion to another
Value = [0 for i in Criteria]
Value[choice] = 1

for criterion in Criteria:
    if criterion != Criteria[choice]:
        try:
            Value[Criteria.index(criterion)] = float(input("How much to you value " + criterion + " over " +
                                                            Criteria[choice] + "? (Enter decimal from 0 to 1) "))
        except ValueError:
            print("Error 100: NaN")
            sys.exit(100)
```

Στιγμιότυπο 10: Ερώτηση χρήστη για το σημαντικότερο κριτήριο και κατάταξη των υπολοίπων σε σχέση με το σημαντικότερο

Το πρόγραμμα μετά θα ρωτήσει το χρήστη ποιο κριτήριο θεωρεί ως σημαντικότερο, και μετά θα τον ρωτήσει πόσο σημαντικά θεωρεί τα υπόλοιπα κριτήρια σε σχέση με το σημαντικότερο κριτήριο. Θα αποθηκεύσει τα ποσοστά (όπως φαίνονται στον πίνακα 5, στήλη “Κατάταξη”) σε μια λίστα με όνομα *Values*.

```
Criteria = list(Data.columns)
Alts = list(Data.index)
# print(Alts, Criteria)

# Build benchmark alternative (the alternative with the worst of each alternative)
benchmark = []
Swings = []

for x in Data.columns:
    benchmark.append(Data[x].max())
    Swings.append(Data[x].min())

# Build swing's fictional alts
Matrix = [[benchmark[i] for i in range(c)] for j in range(r)]
for x in range(c):
    Matrix[x][x] = Swings[x]

# Build swing matrix
SwingData = pd.DataFrame(Matrix, index=Criteria)
print("Our Swing Matrix is:")
print(SwingData)
```

Στιγμιότυπο 11: Επεξεργασία δεδομένων από το αρχείο Excel

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών: Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method σε περιβάλλον Python

Το πρόγραμμα, εν τέλει, υπολογίζει και κανονικοποιεί τα βάρη, όπως φαίνεται στον πίνακα 6, τα τυπώνει, δημιουργεί γράφημα με όνομα “plot_ημερομηνία_ώρα.png” και τα εξάγει σε ένα αρχείο Excel με όνομα “output_ημερομηνία_ώρα.xlsx”.

```
# Normalize weights
total = 0
for value in Value:
    total += value*100

Weight = []
# Calculate weights
for i in range(r):
    Weight.append(Value[i]*100/total)

# Plot results
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.barh(Criteria,Weight)
ax.set_ylabel('Criteria')
ax.set_xlabel('Weights')

# Get date+time for filename
dateTimeObj = datetime.now()
timestampStr = dateTimeObj.strftime("%d-%m-%Y_%H:%M:%S")

# Save to png
plt.savefig('plot_' + timestampStr + '.png',bbox_inches='tight')
print(pd.DataFrame(Weight,Criteria))
pd.DataFrame(Weight,Criteria).to_excel("output_" + timestampStr + '.xlsx', header=False)
```

Στιγμιότυπο 12: Υπολογισμός, εκτύπωση και εξαγωγή βαρών

Εκτέλεση προγράμματος

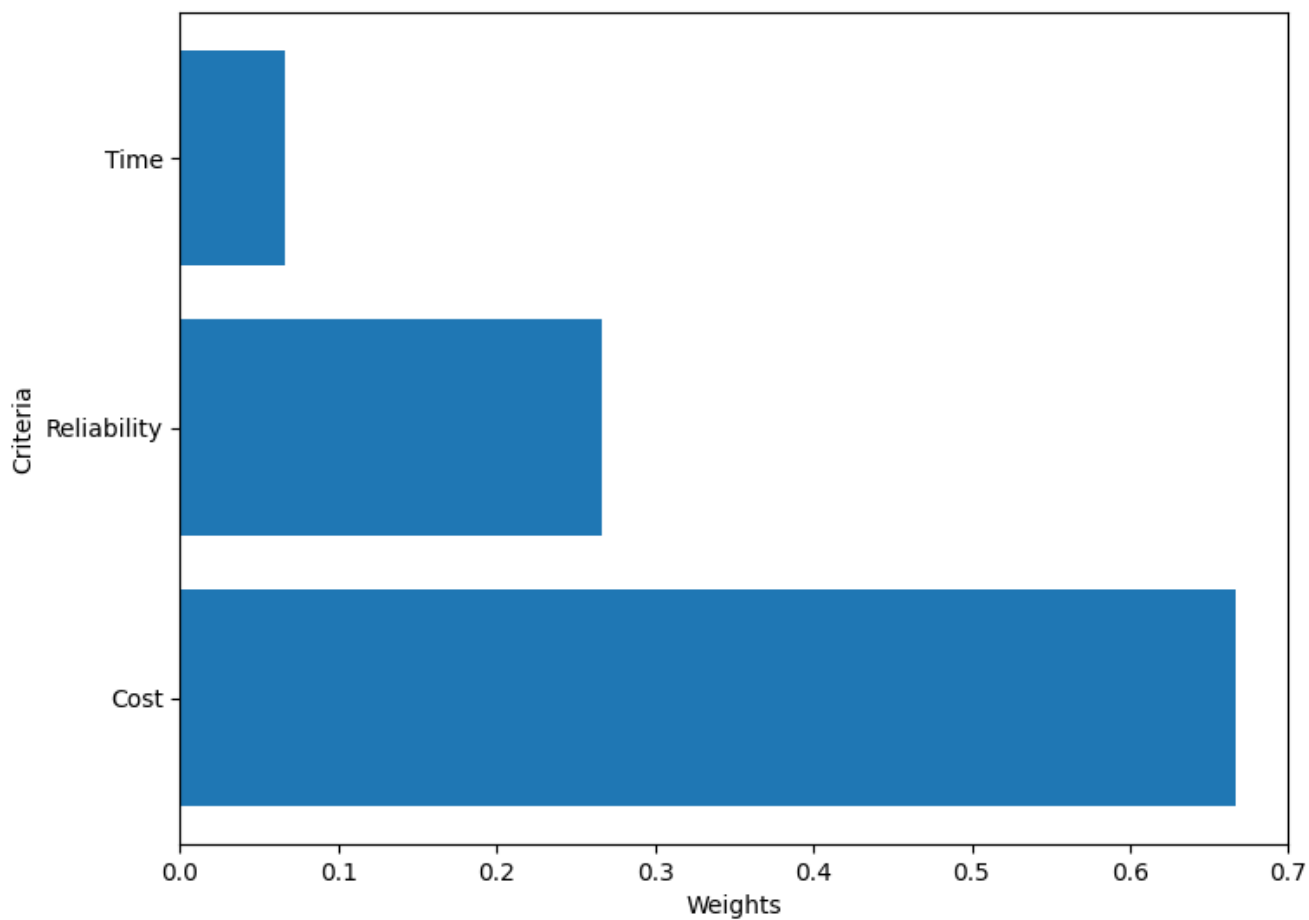
Όπως αναφέραμε παραπάνω, το πρόγραμμα αρχικά διαβάζει και εισάγει δεδομένα από το αρχείο "swingAlts.xlsx" το οποίο πρέπει να βρίσκεται στον ίδιο φάκελο με το εκτελέσιμο αρχείο και μας το τυπώνει στην οθόνη. Μετά δημιουργεί τον πίνακα της μεθόδου swing με την εικονική εναλλακτική benchmark και μας ρωτάει ποιο κριτήριο μας ενδιαφέρει περισσότερο, και του το δίνουμε με έναν ακέραιο 1,2,3,... όπως φαίνεται στον πίνακα. Μετά, μας ρωτάει πόσο μας ενδιαφέρουν τα υπόλοιπα κριτήρια σε σχέση με το σημαντικότερο για εμάς, και μας ζητάει να του δώσουμε έναν δεκαδικό από 0 έως 1, ο οποίος εκφράζει το ποσοστό που μας ενδιαφέρει το εν λόγω κριτήριο. Τέλος, το πρόγραμμα θα μας τυπώσει τα βάρη που προκύπτουν και θα τα εξάγει σε γράφημα και αρχείο Excel (στιγμιότυπα 14 και 15 αντίστοιχα). Η έξοδος του προγράμματος φαίνεται στο στιγμιότυπο 13.

```
panos@panos-pc: ~/Dev/PyCharmProjects/legendary-rotary-phone/Swing
panos@panos-pc:~/Dev/PyCharmProjects/legendary-rotary-phone/Swing$ python3 swingMethod.py
Gtk-Message: 12:27:16.605: Failed to load module "appmenu-gtk-module"
Imported file swingAlts.xlsx!
  Cost  Reliability  Time
A  2200           1    60
B  2500           0    25
C  3000           2    40
Our Swing Matrix is:
      0  1  2
Cost      2200  2  60
Reliability 3000  0  60
Time      3000  2  25
Please pick the criteria that you value most (Enter number 1,2,...): 1
You value Cost most.

How much to you value Reliability over Cost? (Enter decimal from 0 to 1) 0.4
How much to you value Time over Cost? (Enter decimal from 0 to 1) 0.1
      0
Cost      0.666667
Reliability 0.266667
Time      0.066667
/home/panos/.local/lib/python3.8/site-packages/matplotlib/backends/backend_gtk3.py:195: Warn
ing: Source ID 7 was not found when attempting to remove it
  GLib.source_remove(self._idle_draw_id)
panos@panos-pc:~/Dev/PyCharmProjects/legendary-rotary-phone/Swing$
```

Στιγμιότυπο 13: Έξοδος του προγράμματός μας. Τα δεδομένα που εισάγουμε είναι όμοια με τη θεωρητική επίλυση του προβλήματος

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών: Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method σε περιβάλλον Python



Στιγμιότυπο 14: Το διάγραμμα που εξάγει το πρόγραμμα

	A	B
1	Cost	0.66667
2	Reliability	0.26667
3	Time	0.06667

Στιγμιότυπο 15: Το αρχείο Excel που εξάγει το πρόγραμμα

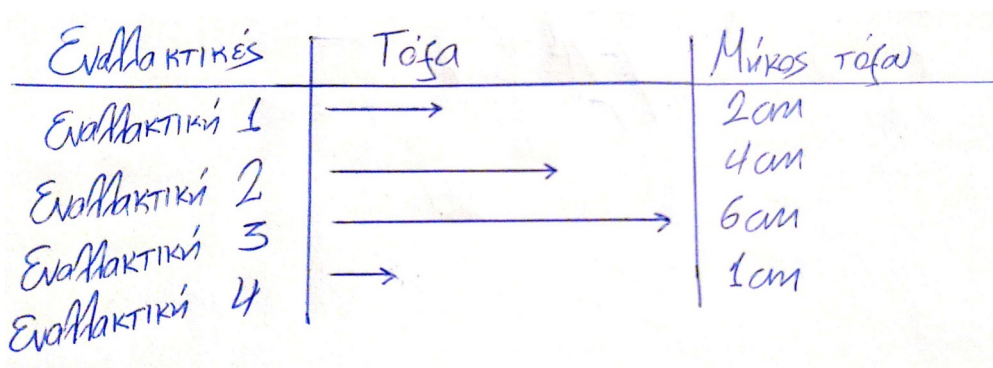
Graphical Weighting Method

Η Μέθοδος

Στη Graphical Weighting Method, ο αποφασίζων καλείται μέσα από ένα πλήθος κριτηρίων ή εναλλακτικών να εξάγει βάρη. Στη συγκεκριμένη μέθοδο, αυτό το επιτυγχάνει με έναν καθαρά γραφικό τρόπο. Συνήθως, ο αποφασίζων σχεδιάζει βελάκια τέτοιου μήκους, όσο τον ενδιαφέρει κάποιο κριτήριο. Τα κριτήρια τα έχουμε το ένα κάτω από το άλλο ώστε να μπορεί ο αποφασίζων να συγκρίνει το μήκος των τόξων εύκολα και γρήγορα. Μόλις σχεδιαστούν όλα τα τόξα, τότε με τη βοήθεια ενός χάρακα μετράμε το μήκος των τόξων, τα κανονικοποιούμε και έτσι προκύπτουν τα βάρη μας. Παρακάτω φαίνεται ένα παράδειγμα, το οποίο εφαρμόστηκε στο χαρτί.

Παράδειγμα

Εφαρμόζουμε την Graphical Weighting Method για 4 εναλλακτικές, όπως φαίνεται στον πίνακα 7. Σχεδιάζουμε τα τόξα ανάλογα πόσο σημαντική θεωρούμε την εναλλακτική και μετράμε το μήκος τους.



Εναλλακτικές	Τόξα	Μήκος τόξου
Εναλλακτική 1	→	2cm
Εναλλακτική 2	→	4cm
Εναλλακτική 3	→	6cm
Εναλλακτική 4	→	1cm

Πίνακας 7: Εφαρμογή της Graphical Weighting Method στο χαρτί

Μετά από αυτό, αθροίζουμε τα μήκη και κανονικοποιούμε, διαιρώντας με το σύνολο των μηκών. Αυτά είναι τα βάρη μας και ολοκληρώνεται η Graphical Weighting Method.

Πίνακας 8: Τα αποτελέσματα της Graphical Weighting Method

Εναλλακτικές	Μήκος τόξου (cm)	Βάρη
Εναλλακτική 1	2	$2/13 = 0.154$ ή 15.4%
Εναλλακτική 2	4	$4/13 = 0.308$ ή 30.8%
Εναλλακτική 3	6	$6/13 = 0.462$ ή 46.2%
Εναλλακτική 4	1	$1/13 = 0.077$ ή 7.7%

Υλοποίηση της μεθόδου σε Python

Περιγραφή

Το πρόγραμμα, αρχικά, διαβάζει από ένα αρχείο Excel, με τη χρήση του *pandas*, τους τίτλους των εναλλακτικών που καλούμαστε να εξάγουμε βάρη. Έπειτα αποθηκεύουμε τους τίτλους σε μια λίστα με όνομα *Criteria*. Η μορφή του αρχείου Excel φαίνεται στο στιγμιότυπο 17.

	A
1	Criteria
2	Criterion 1
3	Criterion 2
4	Criterion 3
5	Criterion 4

Στιγμιότυπο 16: Η μορφή του αρχείου εισόδου μας. Ξεκινάμε στην πρώτη γραμμή με έναν τίτλο και από κάτω βάζουμε τα ονόματα όσων κριτηρίων έχουμε.

```
# import data
Data = pd.read_excel(r'graphicalWeightingData.xlsx', index_col=False)
Criteria = Data.iloc[:,0].tolist()
```

Στιγμιότυπο 17: Ανάγνωση και εισαγωγή του αρχείου. Αρχικά δημιουργούμε ένα data frame με όνομα *Data* και από αυτό παίρνουμε την πρώτη στήλη του και την αποθηκεύουμε στη λίστα *Criteria*.

```
# read graphical weights
Values = []
for criterion in Criteria:
    print("\n\nDraw with '=' the value of "+ criterion+":")
    if not Values == []:
        print("Previous criteria:")
        for i in range(len(Values)):
            print("\n"+Criteria[i] + ": ", end = '')
            for j in range(Values[i]):
                print("=", end = '')
        temp = input("\n"+criterion + ": ")
        Values.append(len(temp))
```

Στιγμιότυπο 18: Ανάγνωση των "τόξων" από το πληκτρολόγιο. Ο χρήστης χρησιμοποιεί κάποιον χαρακτήρα (πχ το '=') και σχεδιάζει τα τόξα όπως θα τα σχεδιάζε στο χαρτί. Το πρόγραμμα τυπώνει τα προηγούμενα τόξα σε κάθε επανάληψη ώστε ο χρήστης να μπορεί να συγκρίνει με ευκολία τα τόξα.

Το πρόγραμμα, μετά, ζητάει από το χρήστη να "σχεδιάσει" τόξα με χρήση κάποιου χαρακτήρα για τα διάφορα κριτήρια που του ορίσαμε. Σε κάθε επανάληψη αποθηκεύουμε το αλφαριθμητικό που αντιστοιχεί στο "τόξο" στη μεταβλητή *temp* και αποθηκεύουμε σε μια λίστα με όνομα *Values* το μήκος αυτού, όπως θα κάναμε, δηλαδή, και στο χαρτί.

Το πρόγραμμα, εν τέλει, υπολογίζει και κανονικοποιεί τα βάρη, όπως φαίνεται στον πίνακα 8, αποθηκεύοντάς τα στη λίστα *Weights*, τα τυπώνει, δημιουργεί γράφημα με όνομα “*plot_ημερομηνία_ώρα.png*” και τα εξαγει σε ένα αρχείο Excel με όνομα “*output_ημερομηνία_ώρα.xlsx*”.

```
# Normalize weights
total = 0
for value in Values:
    total += value

Weights = []
for value in Values:
    Weights.append(value/total)

# Plot results
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.barh(Criteria,Weights)
ax.set_ylabel('Criteria')
ax.set_xlabel('Weights')

# Get date+time for filename
dateTimeObj = datetime.now()
timestampStr = dateTimeObj.strftime("%d-%m-%Y_%H:%M:%S")

# Save to png
plt.savefig('plot_' + timestampStr + '.png',bbox_inches='tight')

# Save to excel & print results
print(pd.DataFrame(Weights,Criteria))
pd.DataFrame(Weights,Criteria).to_excel("output_" + timestampStr + '.xlsx', header=False)
```

Στιγμιότυπο 19: Υπολογισμός και κανονικοποίηση βαρών, εκτύπωση αυτών, δημιουργία γραφικής παράστασης και εξαγωγή σε αρχείο Excel

Εκτέλεση Προγράμματος

Εκτελώντας το πρόγραμμα, αρχικά διαβάζει το αρχείο Excel με όνομα “swingWeightingData.xlsx”. Το πρόγραμμα, τότε, μας ζητάει να σχεδιάσουμε με κάποιο χαρακτήρα (προτείνει τον ‘=’ αλλά δεν είναι δεσμευτικός) το τόξο που αντιστοιχεί στο πρώτο κριτήριο. Μετά, μας επισημαίνει να σχεδιάσουμε το τόξο για το επόμενο κριτήριο, ενώ έχει φροντίσει να μας τυπώσει το προηγούμενο/α τόξο/α ακριβώς από πάνω, ώστε να είναι εύκολο να συγκρίνουμε τα κριτήρια μεταξύ τους. Μόλις έχουμε σχεδιάσει όλα τα τόξα, το πρόγραμμα τότε τυπώνει τα βάρη που προκύπτουν (στιγμιότυπο 20), δημιουργεί το γράφημα, εξάγει τα αποτελέσματα στο αρχείο Excel και κλείνει (στιγμιότυπα 21 και 22).

```
panos@panos-pc: ~/Dev/PyCharmProjects/legendary-rotary-phone/Graphical Weighting Method
panos@panos-pc:~/Dev/PyCharmProjects/legendary-rotary-phone/Graphical Weighting Method$ python3 ./graphical.py
Gtk-Message: 20:09:13.261: Failed to load module "appmenu-gtk-module"

Draw with '=' the value of Criterion 1:
Criterion 1: ==

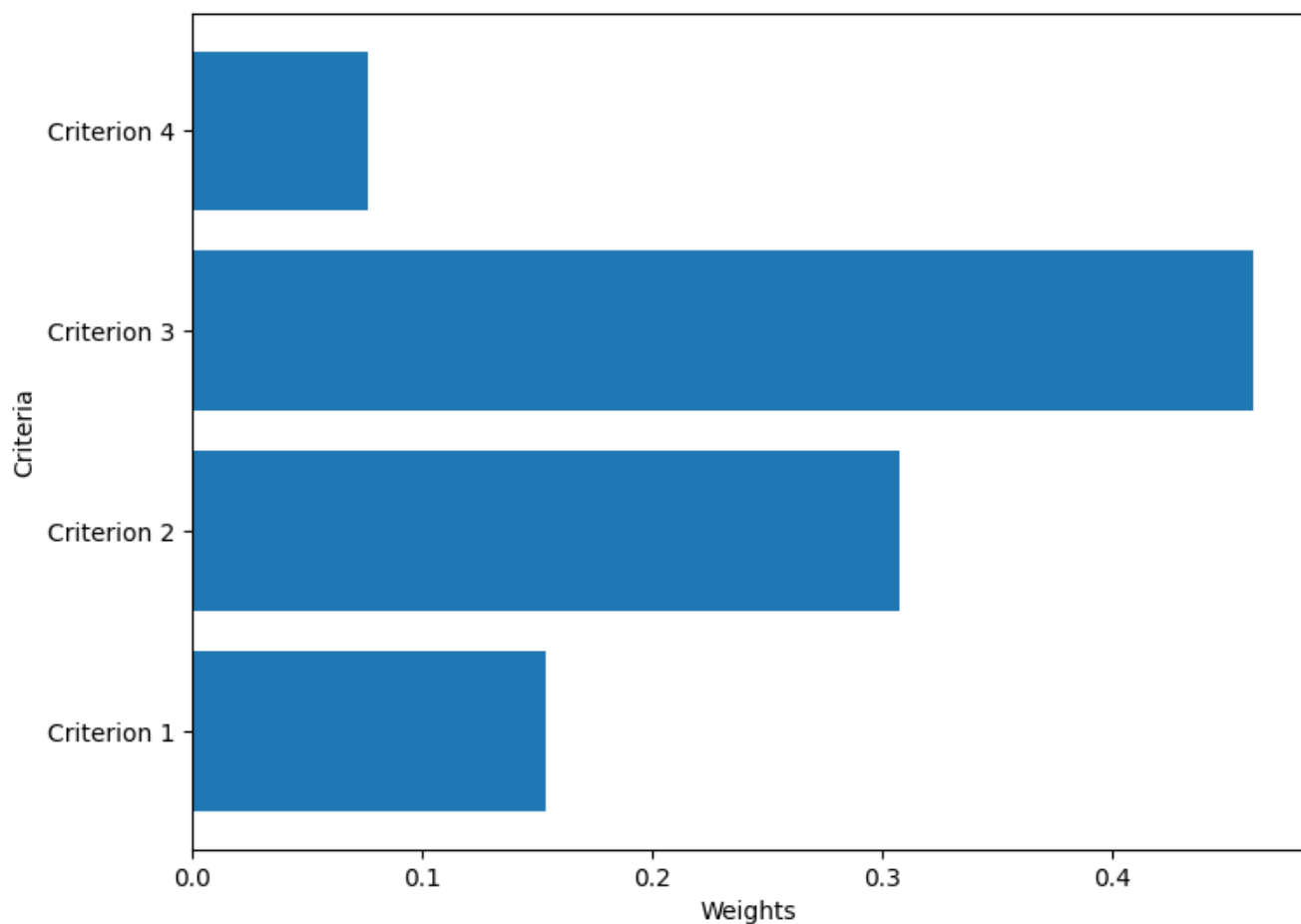
Draw with '=' the value of Criterion 2:
Previous criteria:
Criterion 1: ==
Criterion 2: ====

Draw with '=' the value of Criterion 3:
Previous criteria:
Criterion 1: ==
Criterion 2: ====
Criterion 3: =====

Draw with '=' the value of Criterion 4:
Previous criteria:
Criterion 1: ==
Criterion 2: ====
Criterion 3: =====
Criterion 4: =
0
Criterion 1 0.153846
Criterion 2 0.307692
Criterion 3 0.461538
Criterion 4 0.076923
/home/panos/.local/lib/python3.8/site-packages/matplotlib/backends/backend_gtk3.py:195: Warning: Source ID 7 was not found when attempting to remove it
panos@panos-pc:~/Dev/PyCharmProjects/legendary-rotary-phone/Graphical Weighting Method$
```

Στιγμιότυπο 20: Έξοδος του προγράμματός μας. Τα δεδομένα που εισάγουμε είναι όμοια με τη θεωρητική επίλυση του προβλήματος.

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών: Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method σε περιβάλλον Python



Στιγμιότυπο 21: Το γράφημα που προκύπτει από την εκτέλεση του προγράμματος

	A	B
1	Criterion 1	0.15384615
2	Criterion 2	0.30769231
3	Criterion 3	0.46153846
4	Criterion 4	0.07692308

Στιγμιότυπο 22: Το αρχείο Excel που εξάγει το πρόγραμμα

Delphi Weighting Method

Η Μέθοδος

Στη μέθοδο απόδοσης βαρών delphi εκλέγονται άτομα εξειδικευμένα σε διάφορους τομείς και παραμένουν ανώνυμα. Ύστερα μοιράζονται ερωτηματολόγια σε όλα τα μέλη, συνήθως με την χρήση κανονικού ή ηλεκτρονικού ταχυδρομείου. Αφού συμπληρωθούν τα ερωτηματολόγια από όλους θα συγκεντρωθούν όλες οι απαντήσεις, θα αναλυθούν και θα σταλθούν πάλι πίσω στα εκλεγμένα μέλη. Το κάθε μέλος θα έχει το δικαίωμα να αλλάξει τις απαντήσεις του ανάλογα με τις απαντήσεις που δόθηκαν από άλλα μέλη και θα στείλει τις απαντήσεις πίσω. Τα βήματα αυτά θα συνεχίσουν να επαναλαμβάνονται έως ότου τα αποτελέσματα που στέλνουν τα μέλη έχουν ελάχιστες ή και καθόλου αλλαγές.

Παράδειγμα

Μια εταιρία θέλει να αγοράσει μια αποθήκη και έχει βρει 4 κατάλληλους χώρους (χώρος 1, χώρος 2, χώρος 3, χώρος 4) αλλά για να διαλέξει τον καλύτερο χώρο αποφασίζει ότι θέλει να εκτελέσει την μέθοδο delphi. Έτσι προσλαμβάνει έναν αρχιτέκτονα, έναν πολιτικό μηχανικό και έναν εργολάβο και φροντίζει έτσι κανένα από αυτά τα μέλη να μην γνωρίζει κανένα άλλο μέλος. Ύστερα ζητάει από τον καθένα να αξιολογήσει από το 0 έως το 4 κάθε χώρο, μόλις βγουν τα αποτελέσματα δημιουργείτε ο εξής πίνακας:

Γύρος 1st

	Αρχιτέκτονας	Πολιτικός Μηχανικός	Εργολάβος	Μέσος Όρος
Χώρος 1	0	0	2	0.67
Χώρος 2	4	3	1	2.67
Χώρος 3	2	2	3	2.34
Χώρος 4	4	4	3	3.67

Αφού συγκεντρωθούν και αναλυθούν τα αποτελέσματα του πρώτου γύρου στέλνονται πάλι πίσω στους συμμετέχοντες και απαντάνε πάλι στην ερώτηση, με το εξής αποτέλεσμα:

Γύρος 2nd

	Αρχιτέκτονας	Πολιτικός Μηχανικός	Εργολάβος	Μέσος Όρος	Προηγούμενος Μέσος Όρος
Χώρος 1	0	0	0	0	0.67
Χώρος 2	3	3	2	2.67	2.67
Χώρος 3	2	2	2	2	2.34
Χώρος 4	4	4	4	4	3.67

Γίνεται η ίδια διαδικασία και παράγεται ο παρακάτω πίνακας:

Γύρος 3rd

	Αρχιτέκτονας	Πολιτικός Μηχανικός	Εργολάβος	Μέσος Όρος	Προηγούμενος Μέσος Όρος
Χώρος 1	0	0	0	0	0
Χώρος 2	3	3	2	2.67	2.67
Χώρος 3	2	2	2	2	2
Χώρος 4	4	4	4	4	4

Στον 3^ο γύρο έχουμε συμφωνία με τον προηγούμενο γύρο επομένως η διαδικασία τερματίζεται και τα αποτελέσματα είναι έτοιμα. Τα βάρη προκύπτουν διαιρώντας το εκάστοτε νούμερο του μέσου όρου με το άθροισμα της στήλης αυτής.

Πίνακας 9: Τα αποτελέσματα της Delphi

Χώροι	Βάρη
Χώρος 1	$0/8.67 = 0$ ή 0%
Χώρος 2	$2.67/8.67 = 0.31$ ή 31%
Χώρος 3	$2/8.67 = 0.23$ ή 23%
Χώρος 4	$4/8.67 = 0.46$ ή 46%

Υλοποίηση της μεθόδου σε Python

Περιγραφή

Αρχικά ζητάμε από τον χρήστη να ορίσει το επιθυμητό κατώφλι. Έπειτα, αρχικοποιούμε τις λίστες `answrs` (λίστα με τους μέσους όρους από τον τρέχον γύρο), `oldanswrs` (απαντήσεις προηγούμενου γύρου) και `pinakas_diaforon` (ελέγχει την διαφορά ανάμεσα στις απαντήσεις δυο διαδοχικών γύρων), ορίζουμε ότι δεν υπάρχει συμφωνία (`agrmnt = False`). Έστερα, ξεκινάμε έναν βρόγχο που ελέγχει αν υπάρχει συμφωνία, αν δεν υπάρχει εισάγουμε δεδομένα στο πρόγραμμα μέσω ενός υπολογιστικού φύλλου Excel (η μορφή του αρχείου Excel φαίνεται στο στιγμιότυπο 23) και το αποθηκεύουμε στη μεταβλητή `data`. Στην μεταβλητή `Alts` αποθηκεύουμε μια λίστα με τις εναλλακτικές μας, ύστερα αποθηκεύουμε τις διαστάσεις του πίνακα στις μεταβλητές `r`, `c`.

```
# Threshold.
katofli = float(input("Input threshold (0,05 suggested): "))

# List initiation
oldanswrs = ["nothing old yet"]
answrs = []
pinakas_diaforon = ['There is no diversity yet']

# Agreement check, if it's True the main loop stops.
agrmnt = False

# Main loop. This loop checks the data between rounds, if the data is similar the loop stops.
while agrmnt == False:

    # Importing the excel file.
    data = pd.read_excel(r'data.xlsx', index_col=0)
    Alts = list(data.index)

    # Reading the rows an columns (determines how many questions and how many participants we have).
    r, c = data.shape
```

Στιγμιότυπο 23: Εισαγωγή δεδομένων

	A	B	C	D
1		architect	civil engineer	contractor
2	wh1	1	2	2
3	wh2	1	1	1
4	wh3	0	1	0
5	wh4	1	1	1

Στιγμιότυπο 24: Μορφή του αρχείου που διαβάζει το πρόγραμμα. Στην πρώτη στήλη ξεκινώντας από τη δεύτερη γραμμή εισάγουμε τα ονόματα των εναλλακτικών, ενώ στη πρώτη γραμμή ξεκινώντας από τη δεύτερη στήλη εισάγουμε τα ονόματα των συμμετεχόντων. Τα δεδομένα μπαίνουν από το 0 μέχρι το 4..

Έπειτα αρχικοποιούμε δύο ακόμα μεταβλητές την *diafora* που είναι τα στοιχεία που μπαίνουν στην λίστα *pinakas_diaforon* και την *sumfonia* που είναι για υπεύθυνη για την καταμέτρηση των απαντήσεων που δεν έχουν αλλάξει από τον προηγούμενο γύρο. Ύστερα δημιουργούμε έναν βρόγχο που αναλύει κάθε ερώτηση. Αρχικά αρχικοποιεί μια μεταβλητή *temp* αργότερα τυπώνει την ερώτηση που αναλύει και μετά δημιουργούμε μια καινούρια επανάληψη που τυπώνει την απάντηση κάθε συμμετέχοντα για την συγκεκριμένη ερώτηση και την αθροίζει στην *temp* και μόλις τελειώσει η επανάληψη υπολογίζουμε τον μέσο όρο των απαντήσεων με την χρήση της *temp* και τον αποθηκεύουμε στην μεταβλητή *motemp* που στην συνέχεια βάζουμε στην λίστα *answrs*.

```
# Init of some variables.
diafora = 0
sumfonia = 0

# Analyzing data.
for i in range(r):
    temp = 0
    # Print the current question.
    print("-----er", i+1, "-----", sep='')

    for j in range(c):
        # Print the answer of each user for the specific question.
        print('user', j+1, ': ', data.iloc[i, j], sep='')
        temp += data.iloc[i, j]

    # Average of the answers.
    motemp = temp / c
    answrs.append(motemp)
```

Στιγμιότυπο 25: Επεξεργασία δεδομένων

Μετά αν θα ελέγχουμε αν η λίστα *oldanswrs* έχει την ίδια τιμή που της δώσαμε στην αρχή, αν δεν έχει την ίδια τότε υπολογίζουμε την διαφορά μεταξύ των παλιών και των καινούριων απαντήσεων και την αποθηκεύουμε στην λίστα *pinakas_diaforon*. Ύστερα ελέγχουμε αν η διαφορά πληρεί το ορισμένο από τον χρήστη κατώφλι και αν το πληρεί προσθέτουμε στην *sumfonia* μια μονάδα. Μόλις τελειώσει η επανάληψη που ελέγχει κάθε ερώτηση τυπώνουμε τα αποτελέσματα, καθαρίζουμε τις λίστες *oldanswrs*, *answrs* και *pinakas_diaforon* και περνάμε τα στοιχεία που είχε η *answrs* στην *oldanswrs*. Έπειτα ελέγχουμε αν η *sumfonia* είναι ίση με τον αριθμό των ερωτήσεων που έχουμε τότε αλλάζουμε την *agrmnt* από "False" σε "True" και τερματίζει η επανάληψη ελέγχου συμφωνίας. Αν δεν είναι ίση τότε ζητάμε από τον χρήστη να πατήσει "enter" όταν έχει υποβάλει στο πρόγραμμα τις καινούριες απαντήσεις και είναι έτοιμος.


```
# Exporting data to pinakas_diaforon and checking the agreement.
if oldanswrs[0] != "nothing old yet":
    diafora = oldanswrs[i] - answrs[i]
    pinakas_diaforon.append(abs(round(diafora,2)))

    if pinakas_diaforon[i] < katofli:
        sumfonia += 1

print('-----ANSWERS-----')
print('-The answers from this round are:', answrs)
print('-The answers from the previous round are:', oldanswrs)
print('-The diversity between the last two rounds is:', pinakas_diaforon)

oldanswrs.clear()
oldanswrs = answrs.copy()
answrs.clear()
pinakas_diaforon.clear()

if sumfonia == r:
    agrmnt = True
else:
    input('Press enter to continue...')
```

Στιγμιότυπο 26: Ορισμός πίνακα διαφορών και έλεγχος συμφωνίας.

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών: Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method σε περιβάλλον Python

Το πρόγραμμα, εν τέλει, υπολογίζει και κανονικοποιεί τα βάρη, όπως φαίνεται στον πίνακα 9, τα τυπώνει, δημιουργεί γράφημα με όνομα *"plot_ημερομηνία_ώρα.png"* και τα εξάγει σε ένα αρχείο Excel με όνομα *"output_ημερομηνία_ώρα.xlsx"*.

```
print("-----The weights are-----")

# Normalize weights
total = 0
for item in oldanswrs:
    total += item

Weights=[]
for item in oldanswrs:
    Weights.append(item/total)

for i in range(5):
    print('Er', i, ': ', round(Weights[i], 2), sep='')

# Plot results
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.barh(Alts,Weights)
ax.set_ylabel('Criteria')
ax.set_xlabel('Weights')

# Get date+time for filename
dateTimeObj = datetime.now()
timestampStr = dateTimeObj.strftime("%d-%m-%Y_%H:%M:%S")

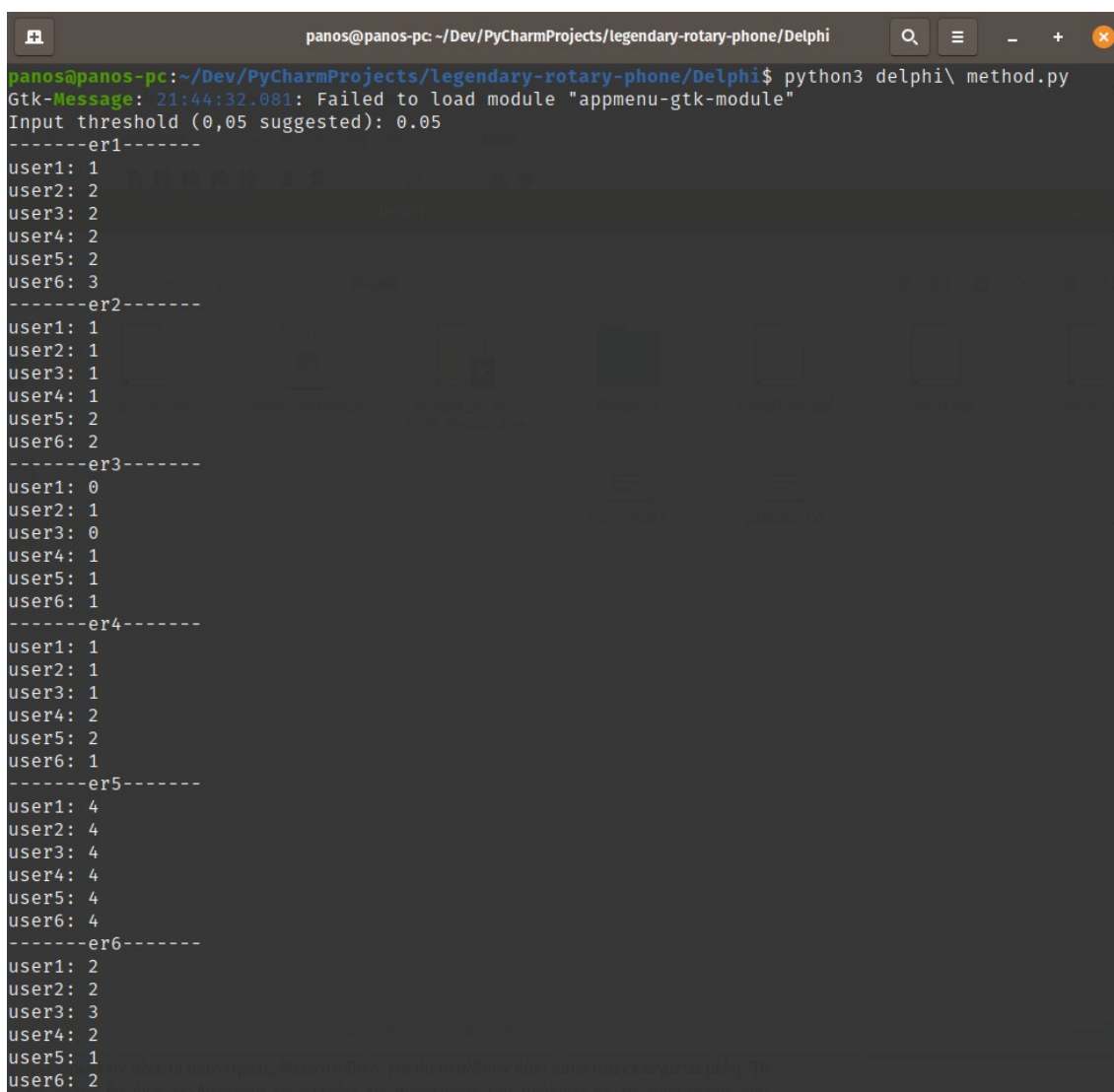
# Save to png
plt.savefig('plot_' + timestampStr + '.png',bbox_inches='tight')

#Save to excel
pd.DataFrame(Weights,Alts).to_excel("output_" + timestampStr + '.xlsx', header=False)
```

Στιγμιότυπο 27: Υπολογισμός, εκτύπωση και εξαγωγή βαρών

Εκτέλεση προγράμματος

Όπως αναφέρθηκε πιο πάνω το πρόγραμμα αρχικά μας ζητάει να ορίζουμε το κατώφλι, ύστερα διαβάζει και εισάγει δεδομένα από το αρχείο "data.xlsx", αφού τυπώσει τα δεδομένα περιμένει μέχρι να ενημερώσουμε το excel και να πατήσουμε έντερ. Μόλις πατήσουμε έντερ το πρόγραμμα θα τυπώσει τα καινούρια δεδομένα, αν τα δεδομένα άλλαξαν θα μας ζητήσει να ενημερώσουμε πάλι τις απαντήσεις και να πατήσουμε έντερ. Αυτή η διαδικασία θα συνεχίσει να επαναλαμβάνεται μέχρι οι καινούριες απαντήσεις που εισάγουμε έχουν αλλάξει ελάχιστα (σύμφωνα με το κατώφλι) ή δεν έχουν αλλάξει καθόλου. Τέλος, το πρόγραμμα θα μας τυπώσει τα βάρη που προκύπτουν και θα τα εξάγει σε γράφημα και αρχείο Excel (στιγμιότυπα 31 και 32 αντίστοιχα). Η έξοδος του προγράμματος φαίνεται στα στιγμιότυπα 28, 29 και 30.



```
panos@panos-pc: ~/Dev/PyCharmProjects/legendary-rotary-phone/Delphi
panos@panos-pc:~/Dev/PyCharmProjects/legendary-rotary-phone/Delphi$ python3 delphi\ method.py
Gtk-Message: 21:44:32.081: Failed to load module "appmenu-gtk-module"
Input threshold (0,05 suggested): 0.05
-----er1-----
user1: 1
user2: 2
user3: 2
user4: 2
user5: 2
user6: 3
-----er2-----
user1: 1
user2: 1
user3: 1
user4: 1
user5: 2
user6: 2
-----er3-----
user1: 0
user2: 1
user3: 0
user4: 1
user5: 1
user6: 1
-----er4-----
user1: 1
user2: 1
user3: 1
user4: 2
user5: 2
user6: 1
-----er5-----
user1: 4
user2: 4
user3: 4
user4: 4
user5: 4
user6: 4
-----er6-----
user1: 2
user2: 2
user3: 3
user4: 2
user5: 1
user6: 2
```

Στιγμιότυπο 28: Η έξοδος του προγράμματος (1/3)

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών: Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method σε περιβάλλον Python

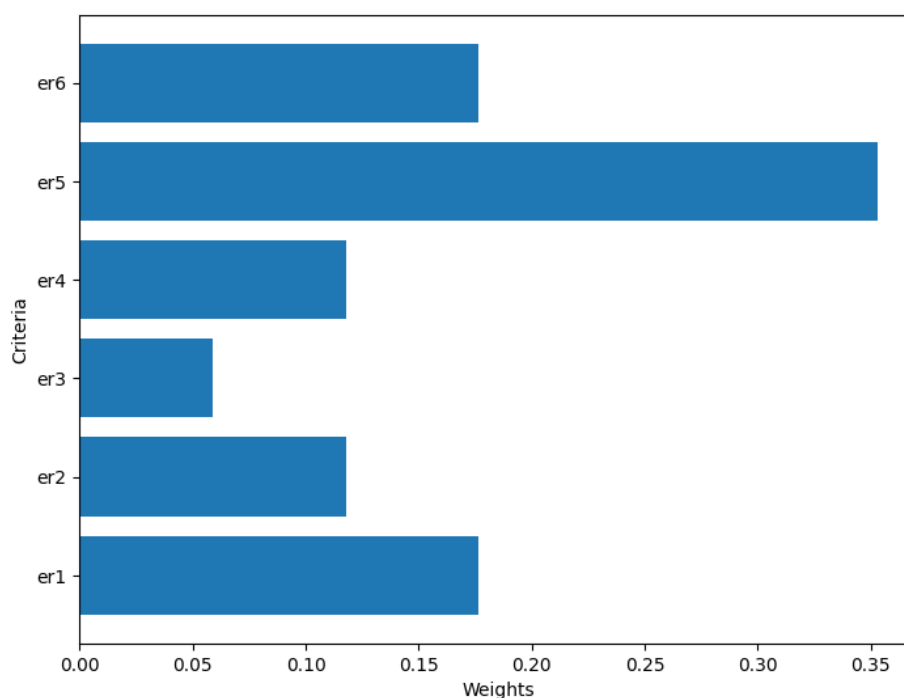
```
panos@panos-pc: ~/Dev/PyCharmProjects/legendary-rotary-phone/Delphi
-----ANSWERS-----
-The answers from this round are: [2.0, 1.3333333333333333, 0.6666666666666666, 1.3333333333333333, 4.0, 2.0]
-The answers from the previous round are: ['nothing old yet']
-The diversity between the last two rounds is: ['There is no diversity yet']
Press enter to continue...
-----er1-----
user1: 1
user2: 2
user3: 2
user4: 2
user5: 2
user6: 3
-----er2-----
user1: 1
user2: 1
user3: 1
user4: 1
user5: 2
user6: 2
-----er3-----
user1: 0
user2: 1
user3: 0
user4: 1
user5: 1
user6: 1
-----er4-----
user1: 1
user2: 1
user3: 1
user4: 2
user5: 2
user6: 1
-----er5-----
user1: 4
user2: 4
user3: 4
user4: 4
user5: 4
user6: 4
-----er6-----
user1: 2
user2: 2
user3: 3
user4: 2
user5: 1
```

Στιγμιότυπο 29: Η έξοδος του προγράμματος (2/3)

Ανάπτυξη συστήματος υλοποίησης μεθόδων απόδοσης βαρών: Graphical Weighting Method, Delphi Method, Fixed Point Scoring και Swing Weighting Method σε περιβάλλον Python

```
-----ANSWERS-----
-The answers from this round are: [2.0, 1.3333333333333333, 0.6666666666666666, 1.3333333333333333, 4.0, 2.0]
-The answers from the previous round are: [2.0, 1.3333333333333333, 0.6666666666666666, 1.3333333333333333, 4.0, 2.0]
-The diversity between the last two rounds is: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
-----The weights are-----
Er0: 0.18
Er1: 0.12
Er2: 0.06
Er3: 0.12
Er4: 0.35
Er5: 0.18
/home/panos/.local/lib/python3.8/site-packages/matplotlib/backends/backend_gtk3.py:195: Warning: Source ID 7 was not
found when attempting to remove it
  Glib.source_remove(self._idle_draw_id)
panos@panos-pc:~/Dev/PyCharmProjects/legendary-rotary-phone/Delphi$
```

Στιγμιότυπο 30: Η έξοδος του προγράμματος (3/3)



Στιγμιότυπο 31: Το γράφημα που εξάγει το πρόγραμμα

	A	B
1	er1	0.17647
2	er2	0.11765
3	er3	0.05882
4	er4	0.11765
5	er5	0.35294
6	er6	0.17647

Στιγμιότυπο 32: Το αρχείο Excel που εξάγει το πρόγραμμα

Βιβλιογραφία

Όλες οι ιστοσελίδες πλοηγήθηκαν και ήταν έγκυρες στις 30 Ιουνίου 2020.

- Weighting Methods and their Effects on Multi-Criteria Decision Making Model Outcomes in Water Resources Management, Noorul Hassan Zardari et al., Springer
- <https://www.napawatersheds.org/img/managed/Document/3456/Hajkowicz2007%20AComparisonOfMultipleCriteriaAnalysisAndUnaid.pdf>
- <https://www.tandfonline.com/doi/pdf/10.1080/713676575>
- https://wiki.ece.cmu.edu/ddl/index.php/Swing_weighting
- https://www.researchgate.net/publication/285402488_231_Using_the_Swing_Weight_Matrix_to_Weight_Multiple_Objectives
- <https://www.youtube.com/watch?v=bHwohMjG9OA>
- https://www.sciencedirect.com/science/article/pii/S0040162505001381?casa_token=Ga9IQ_r1jGAAAAAA:bzeBekX1-s1P-M02QH9lqBths_Y4iCCDdVksFi9WBjqaxq-PMD87DLnP07_JU4k2wt4vUFrhva0
- <https://link.springer.com/article/10.1186/s40201-016-0264-9>
- <https://www.investopedia.com/terms/d/delphi-method.asp>