Εθνικό Μετσόβειο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών

Εργασία στην Python
Time travel project

Προγραμματιστικά Εργαλεία και
Τεχνολογίες για Επιστήμη Δεδομένων

Ονομ/νυμο: Χρήστος Νίκου
email: christosnikou@mail.ntua.gr
ΑΜ: 03400146

27 Φεβρουαρίου 2022

1 Προσέγγιση του προβλήματος

Ο αλγόριθμος που χρησιμοποιηθήκε για την εξαγωγή των δύο ακολουθιών (για $N = 1000$ και $N = 1000000$) είναι ο ίδιος και για τις δύο περιπτώσεις. Η ιδέα που χρησιμοποιηθήκε για την κατασκευή του αλγορίθμου ήταν αρχικά να χωρίσουμε το πρόβλημα σε τρια επιμέρους υποπροβλήματα.

1. Στη δημιουργία της βασικής δομής η οποία θα κρατάει όλες τις απαραίτητες πληροφορίες και θα εκτελεί όλες τις απαραίτητες διεργασίες όπως την ανανέωση υπολοίπου, ανανέωση των δεδομένων για τις υπο κατοχή μετοχές, εκτέλεση μιας συναλλαγής και καταγραφή των συναλλαγών.

2. Στη δημιουργία της στρατηγικής. Δηλαδή, τον τρόπο και τα κριτήρια με ποιον τρόπο θα γίνεται μια συναλλαγή για μια δεδομένη ημέρα.

3. Τα κριτήρια επιλογής των εταιρειών που θα χρησιμοποιηθούν για τις συναλλαγές.

Λόγο του μεγάλου όγκου δεδομένων δεν μπορούμε να επεξεργαστούμε και να διαχειριστούμε όλες τις εταιρείες συγχρόνως και να πετύχουμε έναν αποδοτικό αλγόριθμο. Γι΄ αυτό καταφεύγουμε στην τακτική του (iii). Η γενική ιδέα του αλγορίθμου είναι να διατρέχουμε επαναληπτικά μία - μία τις ημέρες από την αρχική ημερομηνία 1/1/1960 μέχρι την τελική ημερομηνία 10/11/2017 και για μια δεδομένη μέρα βάση της στρατηγικής (ii) να αποφασίζουμε αν θα κάνουμε συναλλαγές ή όχι.

1.1 Επίλυση του Υποπροβλήματος 1 - Η βασική δομή

Για την επίλυση του υποπροβλήματος 1 χρησιμοποιήθηκε η κλάση *tracking* η οποία βρίσκεται στο αρχείο *lib.py* που περιέχει όλες τις απαραίτητες βοηθητικές συναρτήσεις και κλάσεις που χρησιμοποιήθηκαν για το πρόβλημα. Η κλάση *tracking* αποτελεί τη βασική δομή του προβλήματος καθώς κρατάει όλες τις απαραίτητες πληροφορίες για μια δεδομένη μέρα, όπως το υπόλοιπο των χρημάτων που έχουμε στη διάθεσή μας και τις μετοχές που έχουμε στην κατοχή μας. Επίσης, μέσω αυτής της κλάσης υλοποιούνται και όλες οι πιθανές συναλλαγές που γίνονται μέσα στη μέρα.

Πιο αναλυτικά, στο αρχείο του βασικού προγράμματος *mainscript.py* η μεταβλητή τύπου *tracking* αρχικοποιείται μέσω της εντολής.

```
time_traveler = lib.tracking(portfolio)
```

Πριν εξηγήσουμε τις βασικές λειτουργίες της κλάσης *tracking* να πούμε δυο λόγια για την μεταβλητή αρχικοποιήσης *portfolio*. Η μεταβλητή *portfolio* είναι ένα αντικείμενο τύπου *dictionary* με κλειδιά (keys) τις ονομασίες των εταιρειών που έχουν επιλεχθεί μέσω του υποπροβλήματος 3 ή από τον χρήστη (βλ. παράγραφο 3) και τιμές (values) όλες τις απαραίτητες πληροφορίες που χρειαζόμαστε να έχουμε για την κάθε εταιρεία που χρησιμοποιούμε για τις συναλλαγές. Πιο συγκεκριμένα, αν πούμε π.χ. ότι η εταιρεία είναι η AAPL τότε η τιμή time_traveler.portfolio['AAPL'] είναι και αυτή ένα *dictionary* με ζεύγη (κλειδί, τιμή) που ορίζονται ως εξής:

Πίνακας 1: Οι τιμές της μεταβλητής *time_traveler.portfolio['AAPL']*

Κλειδί	**Τιμή**
number of stocks	Πλήθος μετοχών
close price	Τιμή κλεισίματος (για δεδομένη μέρα)
bought	Πλήθος αγορασμένων μετοχών (για δεδομένη μέρα)
sold	Πλήθος μετοχών που πουλήθηκαν (για δεδομένη μέρα)
action	Ενέργεια (π.χ. buy low)
intra	Αν έγινε intra day ή όχι (False, True)
moves	Πλήθος συνολικών κινήσεων

Για παράδειγμα η τιμή time_traveler.portfolio['AAPL']['number of stocks'] αντιστοιχεί στο πλήθος των μετοχών που έχουμε στην κατοχή μας από την εταιρεία της APPLE.

Η τιμή time_traveler.portfolio['AAPL']['close price'] περιέχει την τιμή κλείσιματος της μετοχής της APPLE για μια δεδομένη ημέρα. Την τιμή αυτή τη χρειαζόμαστε για να υπολογίζουμε το *portfolio* στο τέλος της ημέρας. Οι τιμές time_traveler.portfolio['AAPL']['bought'], portfolio['AAPL]['sold'] περιέχουν το πλήθος των μετοχών που αγοράστηκαν και πουλήθηκαν από την APPLE στο τέλος της ημέρας. Η λογική τιμή portfolio['AAPL]['action'] περιέχει το είδος της κίνησης που έγινε για την APPLE μέσα στην ημέρα (π.χ.buy low, sell high, sell open κ.τ.λ).

Η λογική τιμή time_traveler.portfolio['AAPL']['intra'] περιέχει την ένδειξη αν έγινε ή όχι κάποιο intra day trading για τις μετοχές της APPLE.

1.2 Ανανέωση πληροφορίας - Υπολογισμός των Balance, Portfolio

Έχοντας περιγράψει πως μέσω της κλάσης *tracking* μπορούμε να κρατάμε τις βασικές πληροφορίες για τις μετοχές που έχουμε στην κατοχή μας τώρα είναι σχετικά απλό να υπολογίσουμε τις τιμές του balance και του portfolio στο τέλος μιας δεδομένης ημέρας. Αρχικά, πριν τον επαναληπτικό βρόγχο οι εταιρείες που έχουν επιλεχθεί αποθηκεύονται σε μια μεταβλητή *companies*. Ύστερα, μέσω της συνάρτησης *initialize_portfolio* που βρίσκεται στη βιβλιοθήκη *lib.py* αρχικοποιείται το *dictionary* με τις εξής τιμές (π.χ. για την εταιρεία της APPLE): time_traveler.portfolio['AAPL']['number of stocks'] = 0, time_traveler.portfolio['AAPL']['close price'] = 0, time_traveler.portfolio['AAPL']['bought'] = 0, time_traveler.portfolio['AAPL]['sold'] = 0, time_traveler.portfolio['AAPL]['action'] = None και time_traveler.portfolio['AAPL']['intra'] = False.

Τώρα, μετά το τέλος της ημέρας (δηλαδή στο τέλος της επανάληψης) η κλάση *tracking* υπλοποιεί τις συναρτήσεις *cal_number_of_stocks* και *cal_portfolio* οι οποίες ανανεώνουν το συνολικό πλήθος των μετοχών που έχουμε στην κατοχή μας μέσων των τιμών time_traveler.portfolio[company]['bought'], time_traveler.portfolio[company]['sold'] και υπολογίζουν τις τιμές των portfolio, balance αντίστοιχα.

Μετά την ανανέωση η κλάση *tracking* υλοποιεί τη συνάρτηση *clear_daily_stock_info* η οποία με τη σειρά της επαναφέρει τις μεταβλητές time_traveler.portfolio[company]['close price'], time_traveler.portfolio[company]['bought'], time_traveler.portfolio[company]['action'] και τη μεταβλητή time_traveler.portfolio[company]['intra'] στην αρχική τους κατάσταση, όπου η μεταβλητή company διατρέχει όλες τις εταιρείες που έχουμε στην κατοχή μας. Η μεταβλητή *money* της κλάσης

tracking κρατάει το υπόλοιπο του διαθέσιμου ποσού χρημάτων που έχουμε ανά χρονική στιγμή και αρχικοποιείται στην τιμή 1 με το που καλείται ο constructor της κλάσης.

1.3 Έλεγχος των περιορισμών

Ο έλεγχος των περιορισμών για το πλήθος των μετοχών που μπορούμε να πούλάμε ή να αγοράζουμε για μια συγκεκριμένη μετοχή γίνεται μέσω των συναρτήσεων *check_buy* και *check_sell* που βρίσκονται στο αρχείο των συναρτήσεων *lib.py*. Πιο συγκεκριμένα, η συνάρτηση *check_buy* δέχεται ως ορίσματα τις εξής μεταβλητές: *volume, possessed_stocks, bought, num_stocks, money, price* και επιστρέφει την τιμή True αν επιτρέπεται η συναλλαγή αλλιώς επιστρέφει την ένδειξη False.

Η μεταβλητή *volume* αντιστοιχεί στον όγκο της μετοχής που θέλουμε να αγοράσουμε για μια δεδομένη μέρα, η μεταβλητή *possessed_stocks* περιέχει το πλήθος των μετοχών που έχουμε από τη συγκεκριμένη εταιρεία που θέλουμε να αγοράσουμε. Η μεταβλητή *bought* στο πλήθος των μετοχών που έχουμε αγοράσει εκείνη τη δεδομένη μέρα απ΄ την εταιρεία που θέλουμε να αγοράσουμε. Η μεταβλητή *num_stocks* το πλήθος των μετοχών που θέλουμε να αγοράσουμε, *money* τα χρήματα που έχουμε στη διαθεσή μας και *price* το κόστος της μετοχής τη δεδομένη χρονική στιγμή. Αντίστοιχη ερμηνεία έχουν και τα ορίσματα *volume, possessed_stocks, sold, num_stocks, bought* της συνάρτησης *check_sell*.

1.4 Οι βασικές κινήσεις της κλάσης *tracking*

Οι βασικές κινήσεις που μπορούμε να κάνουμε κατά σειρά στη διάρκεια μιας ημέρας είναι οι

$$\{\text{buy open}, \text{sell open}\} \ll \{\text{buy low}, \text{sell high}\} \ll \{\text{buy close}, \text{sell close}\}. \tag{1.1}$$

Οι κινήσεις αυτές υλοποιούνται μέσω των συναρτήσεων *buy_open, sell_open, buy_low, sell_high, buy_close, sell_close* της κλάσης *tracking*. Πριν εφαρμοστεί μια απ΄ τις προηγούμενες κινήσεις ελέγχεται πρώτα η εγκυρότητά της μέσω των συναρτήσεων *check_buy* και *check_sell*. Μετά την υλοποίησης της κίνησης, ανανεώνεται το υπολοιπόμενο διαθέσιμο ποσό χρημάτων. Στις μεταβλητές time_traveler.portfolio[company]['sold'], time_traveler.portfolio[company]['bought'] προστείθονται οι μετοχές που πουλήθηκαν/αγοράστηκαν. Επίσης, σε δύο λίστες εν ονόματι *moves, time_stocktaking* οι οποίες αρχικοκοποιούνται όταν καλείται ο constructor της κλάσης *tracking* αποθηκεύονται οι κινήσεις που γίνανε μέσα στην ημέρα αλλά και το συνολικό ποσό μετά το τέλος της ημέρας (αποτίμηση). Αυτές οι λίστες χρησιμοποιούνται στο τέλος για την εξαγωγή των αρχείων *small.txt, large.txt* αλλά και για το διάγραμμα αποτίμησης. Τέλος, μέσω της κλάσης *tracking* υλοποιούνται και οι κινήσεις που αφορούν το intra day trading. Πιο συγκεκριμένα, η κλάση υλοποιεί τις τέσσερις κινήσεις τύπου intra day που μας συμφέρει να εκτελούμε. Αυτές είναι οι *intra_high_close, intra_open_low, intra_open_high* και *intra_close_low*. Για παράδειγμα, η συνάρτηση *intra_high_close* αντιστοιχεί στην κίνηση που πουλάμε μετοχές στο sell high και αγοράζουμε στο close την ίδια μέρα. Αντίστοιχη είναι και η ερμηνεία των υπόλοιπων συναρτήσεων.

1.5 Επίλυση του Υποπροβλήματος 2 - Η στρατηγική

Έχοντας ως δεδομένο το σύνολο των εταιρειών με το οποίο θα γίνουν οι συναλλαγές, πριν ξεκινήσει η επαναληπτική διαδικασία προσπέλασης όλων των ημερών από την 1/1/1960 εως την 10/11/2017

πρώτα αρχικοποιούμε το dictionary με τις εταιρείες που έχουν επιλεχθεί. Αυτό γίνεται μέσω της εντολής

```
portfolio = lib.initialize_portfolio(companies)
```

Η συνάρτηση *initialize_portfolio* δέχεται ως είσοδο τη λίστα με τα ονομάτα των εταιρειών και για κάθε μια απ΄ αυτές αρχικοποιεί τις μεταβλητές του Πίνακα 1. ΄Υστερα, η όλη διαδικασία των συναλλαγών γίνεται μέσω της συνάρτησης *trading_loop*. Πριν την επαναληπτική διαδικασία αρχικοποιείται η μεταβλητή *time_traveler* τύπου *tracking*. Στη συνέχεια μέσω των εντολών

```
df = generate_dataframe(companies)
df.sort_values(by = ['Date', 'Max_val'], ascending = [True, False], inplace = True)
```

διαβάζονται τα δεδομένα των εταιρείων από τα αντίστοιχα αρχεία *txt* και τοποθεντούνται ταξινομημένα ως προς τις ημερομηνίες σε αύξουσα σειρά σε ένα *DataFrame*. Επιπλέον, η συνάρτηση *generate_dataframe* δημιουργεί τις εξής επιπρόσθετες πληροφορίες για κάθε μια απ΄ τις επιλεγμένες εταιρείες ξεχωριστά.

- Τη μεταβλητή *Max_val* η οποία περιέχει τη μέγιστη τιμή της στήλης High.
- Τη μεταβλητή *Min_val* η οποία περιέχει την ελάχιστη τιμή της στήλης Low.
- Τις μεταβλητές *open_low, close_low, high_open, high_close* οι οποίες περιέχουν τη μέγιστη διαφορά μεταξύ των στηλών Open - Low, Close - Low, High - Open, High - Close αντιστοίχως.
- Στις μεταβλητές *mean_high, mean_low* αποθηκεύονται μέσες τιμές των τιμών High και Low ανά έτος.
- Επίσης, στη μεταβλητή *life_span* αποθηκεύουμαι για κάθε ημέρα το ποσοστό ζωής της εταιρείας που έχουμε διανύσει.

Για κάθε μέρα εκτελούνται οι εξής κινήσεις κατά σειρά

(i) Πρώτα ελέγχουμε όλες τις εταιρείες ανά δύο ξεκινώντας από αυτές με το μεγαλύτερο *Max_val* και ελέγχουμε αν το διαθέσιμο ποσό που έχουμε στην κατοχή μας μαζί με την αξία των μετοχών που έχουμε απ΄ την εταιρεία που έχει το μικρότερο *Max_val* επαρκεί για να αγοράσουμε όσο περισσότερες μετοχές μπορούμε απ΄ την εταιρεία με το μεγαλύτερο *Max_val*. Με αυτόν τον τρόπο ξεκινάμε απ΄ τις εταιρείες με χαμηλή τελική τιμή και σιγά-σιγά αγοράζουμε μετοχές οπότε μπορούμε από εταιρείες με υψηλή τελική τιμή στο High.

(ii) Ελέγχοντας πάντα με λογικές μεταβλητές αν τηρείται η χρονική διάταξη της (1.1) μετά ελέγχουμε για κάθε εταιρεία ξεχωριστά αν είναι καλή μέρα να κάνουμε buy_low ή sell_high. Το κριτήριο για το buy_low μεταφράζεται μέσω της ανίσωσης

$$\text{Low\_value} \leq x \cdot \textit{mean\_low},$$

όπου x είναι ένας πολλαπλασιαστικός παράγοντας (υπερπαράμετρος) ο οποίος επιλέγεται στην αρχή του προγράμματος. Να σημειώσουμε εδώ ότι για την υπερπαράμετρο x χρησιμοποιούμε δύο τιμές, μια τιμή (π.χ. 1.4) όταν ο χρόνος ζωής της εκάστοτε εταιρείας είναι < 0.70 και μια τιμή (π.χ. 1.1) αν ο χρόνος ζωής είναι > 0.7. Με αυτόν τον τρόπο αγοράζουμε πιο ¨έντονα¨

μετοχές εως το 70% του χρόνου ζωής της εταιρείας και αγοράζουμε λιγότερες μετοχές όταν ξεπεράσουμε το 70% του χρόνου ζωής της εταιρείας. Δεδομένουμε ότι έχουμε επιλέξει εταιρείες που η τιμή των μετοχών τους αυξάνεται με την πάροδο του χρόνου είναι λογικό να θέλουμε αρχικά να μαζέψουμε όσες περισσότερες μετοχές γίνεται αρχικά και να τις κρατήσουμε για τις πουλήσουμε στο τέλος. Φυσικά, η μεταβλητή x, καθώς και το ποσοστό απ΄ το οποίο και πέρα θα πουλάμε λιγότερο είναι υπερπαράμετροι και διαφορετικές τιμές δίνουν και αρκετά διαφορετικά αποτελέσματα. Αντίθετα, το κριτήριο για sell_high εκφράζεται μέσω της ανίσωσης

$$\text{High\_value} \geq y \cdot \textit{mean\_high}$$

όπου y είναι μια υπερπαράμετρος αντίστοιχη της x. Εδώ σε αντίθεση με την περίπτωση του buy low θέλουμε σε αρχικό στάδιο να πουλάμε λίγες μετοχές και σε τελικό στάδιο να πουλάμε περισσότερες μετοχές. Το να πουλάμε λίγες μετοχές στην αρχή το πετυχαίνουμε θέτοντας μια σχετικά υψηλή τιμή στην υπερπαράμετρο y, π.χ. $y = 1.4$ ενώ το να πουλάμε αρκετές μετοχές προς το τέλος το πετυχαίνουμε θέτοντας μια μικρότερη τιμή, π.χ. $y = 1.1$.

(iii) Αν για μια εταιρεία μέσα στην ημέρα δεν έχει πάρει μέρος σε καμιά απ΄ τις παραπάνω ενέργειες τότε ελέγχεται αν μπορεί να γίνει intra trade για αυτή τη μετοχή. Το ποιό τύπου intra trade που θα γίνει καθορίζεται μέσω των μεταβλητών *open_low, close_low, high_open, high_close* που ορίστηκαν προηγουμένως. Πάλι σε αυτό το σημείο υπάρχει ένας συντελεστής z ο οποίος ¨ελέγχει¨ αν θα γίνει intra trade ή όχι.

(iv) Τέλος, υπάρχει μια περίοδος (υπερπαράμετρος, π.χ. 500 ημέρες) πριν το τέλος της επανάληψης που γίνεται μόνο intra day trading. Η λογική σε αυτό το σημείο είναι ότι αφού θα έχουμε μαζέψει αρκετές μετοχές απ΄ τις εταιρείες τότε αναμένουμε να επωφεληθούμε αρκετά.

1.6 Επίλυση Υποπροβλήματος 3 - Κριτήρια επιλογής εταιρειών

Γενικά η επιλογή των εταιρειών έγινε μετά από μερικές δοκιμές και συνδυασμούς. Ένα αρχικό κριτήριο για την απαλοιφή των περισσότερων εταιρειών είναι η διαφορά της Close τιμής της εταιρείας την τελευταία ημέρα με την τιμή Low τιμή της εταιρείας την πρώτη μέρα. Παρακάτω βλέπουμε τις πρώτες πέντε γραμμές των εταιρειών με τη μεγαλύτερη διαφορά που περιγράψαμε προηγουμένως.

Πίνακας 2: Οι πρώτες 5 γραμμές του dataframe με βάση το Close - Low

Date	**Open**	**High**	**Low**	**Close**	ID	Close - Low
2017-11-10	275281.00	275660.00	273212.760	275660.00	BPK-A	275370.0000
2017-11-10	4260.00	4328.600	4201.050	4201.05	SEB	4079.9400
2017-11-10	3299.00	3308.510	3239.110	3289.64	NVR	2494.6200
2017-11-10	1700.00	1709.520	1682.690	1697.25	PCLN	1675.0400
2017-11-10	1126.10	1131.750	1124.060	1125.35	AMZN	1123.6400

Για τα δύο αρχεία *large.txt, small.txt* χρησιμοποιήθηκαν μόνο εταιρείες απ΄ τις πρώτες 200 γραμμές του παραπάνω DataFrame. Ο λόγος που έγινε αυτό είναι για να εξασφαλίσουμε ότι οι εταιρείες που

θα επιλεχθούν θα έχουν όσο το δυνατόν μεγαλύτερη αύξηση στην τιμή της μετοχής του απ΄ την αρχή της ζωής τους μέχρι την τελική ημερομηνία.

Ένα δεύτερο κριτήριο επιλογής εταιρειών είναι το *variance* το οποίο υπολογίζεται μέσω της

$$variance = \frac{1}{N}\sum_{i}^{N}\Big(\text{High}[i] - \text{Low}[i]\Big),$$

όπου το άθροισμα λαμβάνεται πάνω από όλες τις ημέρες ζωής της εταιρείας και $\text{High}[i]$, $\text{Low}[i]$ αντιστοιχούν στην τιμή του High και την ημέρα i αντιστοίχως. Συνεπώς, με αυτό το κριτήριο οι καλές εταιρείες είναι αυτές που έχουν το μεγαλύτερο variance. Αυτό δείχνει ότι έχουν μεγάλες αυξομειώσεις κατά τη διάρκεια της ημέρας και επομένως είναι καλές για intra trade. Παρακάτω βλέπουμε τις πρώτες 5 γραμμές του DataFrame με τις εταιρείες που έχουν το μεγαλύτερο *variance*.

Πίνακας 3: Οι πρώτες 5 γραμμές του dataframe με βάση το *variance*

Open	**High**	**Low**	**Close**	ID	Close - Low	Variance
275281.00	275660.00	273212.760	275660.00	BPK-A	275370	979.341077
4260.00	4328.600	4201.050	4201.05	SEB	4079	53.220864
3299.00	3308.510	3239.110	3289.64	NVR	2494	25.822941
1700.00	1709.520	1682.690	1697.25	PCLN	1675	13.664436
1126.10	1131.750	1124.060	1125.35	CABO	1123	12.337800

Τέλος, ένα άλλο σημαντικό κριτήριο για την επιλογή των εταιρειών είναι το Volume και ο χρόνος ζωής των εταιρειών. Ιδανικά, θέλουμε οι εταιρείες που θα επιλέξουμε να έχουν υψηλό Volume καθ΄ όλη τη διάρκεια ζωής τους ώστε να μπορούμε να κάνουμε όσον το δυνατόν περισσότερες συναλλαγές γίνεται. Επίσης, σε αρχικό στάδιο θέλουμε να επιλέξουμε κάποιες εταιρείες πάνω στις οποίες θα ¨χτίσουμε¨ τις συναλλαγές μας. Δηλαδή, θέλουμε να βρούμε εταιρείες οι οποίες έχουν θετικό Variance και έχουν μεγάλο χρόνο ζωής.

2 Τα αποτελέσματα

2.1 *small.txt*

Για το αρχείο small.txt οι εταιρείες που χρησιμοποιήθηκαν είναι οι

```
companies = {'GE', 'AMZN', 'AAPL'}
```

Με την παρακάτω εντολή

```
>mainscript.py 1000
```

τρέχουμε στο command line το πρόγραμμα μέγιστο πλήθος κινήσεων $N = 1.000$. Ο συνδυασμός των παραπάνω εταιρειών οδηγεί σε κέρδος $4,670,718.898$ δολαρίων. Παρακάτω το βλέπουμε το κέρδος, το πλήθος των συναλλαγών αλλά και το διάγραμμα της αποτίμησης και του διαθέσιμου ποσού χρημάτων ανά ημέρα.

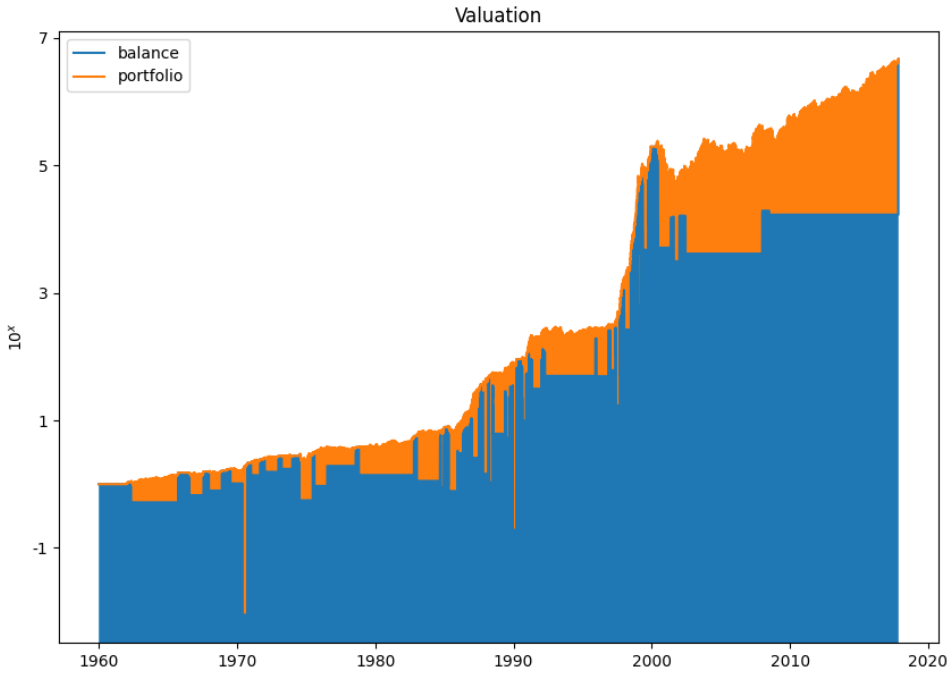
```
 ----> The Time Travel Project 2022 <----

Summary of results:
```

```
- Upper bound of total transactions:  1000
- Default Companies:  {'GE', 'AAPL', 'AMZN'}
- Total transactions:  936
- Money earned:  4670718.89886

Run time:  Hour(s):0.0, Minute(s):0.0, Seconds:56
```

Σχήμα 1: Διάγραμμα αποτίμησης, $N = 1000$



2.2 *large.txt*

Για το αρχείο large.txt οι εταιρείες που χρησιμοποιήθηκαν είναι οι

```
companies = {'TSLA', 'AMZN', 'GOOG', 'GS', 'FB', 'PCLN', 'AAPL',
                'BIIB', 'CMG', 'GE','GOOGL', 'ANET'}
```

Με την παρακάτω εντολή

```
>mainscript.py 1000000
```

τρέχουμε στο command line το πρόγραμμα μέγιστο πλήθος κινήσεων $N = 1.000.000$. Ο συνδυασμός των παραπάνω εταιρειών οδηγεί σε κέρδος 8, 118, 150, 718.85645 δολαρίων. Παρακάτω βλέπουμε το κέρδος, το πλήθος των συναλλαγών αλλά και το διάγραμμα της αποτίμησης και του διαθέσιμου ποσού χρημάτων ανά ημέρα.

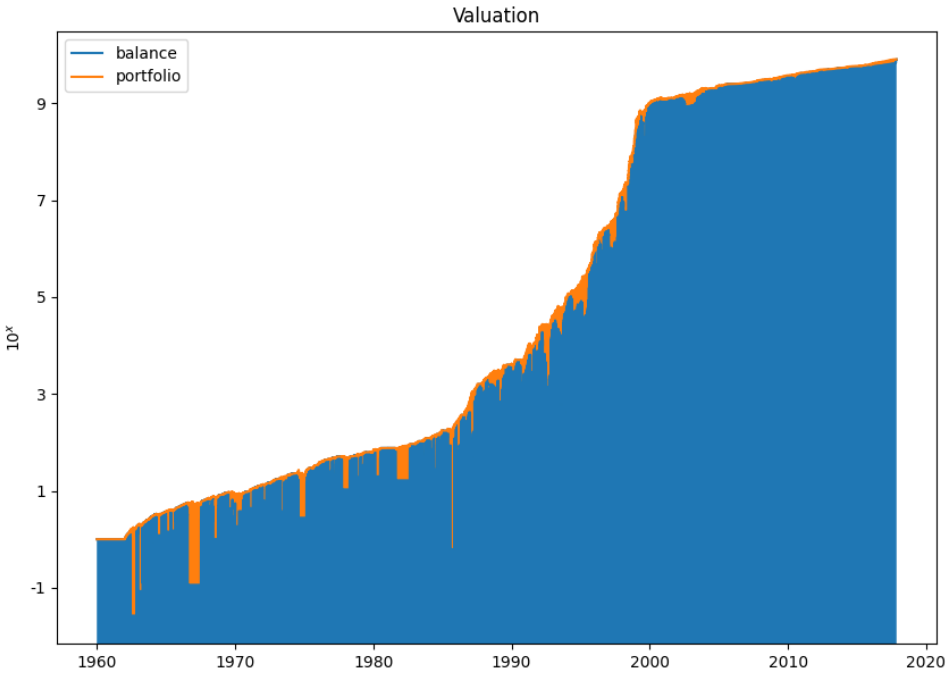
```
 ----> The Time Travel Project 2022 <----

Summary of results:
- Upper bound of total transactions:  1000000
- Default Companies:  {'FB', 'GOOG', 'GE', 'PCLN', 'GOOGL', 'BIIB', 'GS', 'TSLA', 'ANET'
    , 'AAPL', 'CMG', 'AMZN'}
- Total transactions:  32616
```

```
- Money earned:  8118150718.85645

Run time:  Hour(s):0.0, Minute(s):2.0, Seconds:47
```

Σχήμα 2: Διάγραμμα αποτίμησης, $N = 1000000$



3 Εκτέλεση από το Command line

Σε αυτό το σημείο δίνουμε κάποιες οδηγίες για την εκτέλεση του βασικού αρχείου *mainscript.py* από το command line. Αρχικά με την εντολή

```
>mainscript.py 1000
```

εκτελούμε τον αλγόριθμο με τις default εταιρείες (GE, AAPL, AMZN) που αντιστοιχούν στο πλήθος κινήσεων $N = 1.000$. Με την εντολή

```
>mainscript.py 1000000
```

εκτελούμε τον αλγόριθμο με τις default εταιρείες που αντιστοιχούν στο πλήθος κινήσεων $N = 1.000.000$. Απο κει και πέρα αν για παράδειγμα θέλουμε να εμφανίσουμε το πλήθος των κινήσεων που γίνανε μπορούμε να γράψουμε

```
>mainscript.py 1000 PRINT TRUE
```

για το $N = 1.000$ και

```
>mainscript.py 1000000 PRINT TRUE
```

για το $N = 1.000.000$. Για παράδειγμα αν εκτελέσουμε την πρώτη εντολή τότε στις τελευταίες γραμμές θα έχουμε την παρακάτω εικόνα.

```
2008-07-03 buy-low GE 32
2008-07-07 buy-low GE 64
2008-07-09 sell-high GE 127
2008-07-09 buy-close GE 127
2017-11-06 sell-high AMZN 4121
2017-11-06 sell-high GE 127
2017-11-06 sell-high AAPL 77


Summary of results:
- Upper bound of total transactions:  1000
- Default Companies:  {'AMZN', 'GE', 'AAPL'}
- Total transactions:  936
- Money earned:  4670718.89886

Run time:  Hour(s):0.0, Minute(s):0.0, Seconds:48
```

Αν τώρα θέλουμε να αποθηκεύσουμε τα αρχεία *small.txt, large.txt* γράφουμε

```
>mainscript.py 1000 SAVEFILE TRUE
```

και παράγεται το αρχείο *small.txt* στο ίδιο directory που βρίσκονται και τα αρχεία *mainscript.py, lib.py*, και για το αρχείο *large.txt* γράφουμε

```
>mainscript.py 1000000 SAVEFILE TRUE
```

Να σημειώσουμε σε αυτό το σημείο ότι για να τρέξει ο αλγόριθμος θα χρειαστεί ο φάκελος των δεδομένων Stocks να βρίσκεται στο ίδιο directory με τα αρχεία *mainscript.py, lib.py*. Τώρα, αν θέλουμε να χρησιμοποιήσουμε άλλες εταιρείες, π.χ. τις GE, AMZN, FB, NFLX, AAPL, TSLA στις $N = 1000000$ μπορούμε να γράψουμε

```
>mainscript.py 1000000 COMPANIES 6 GE AMZN FB NFLX AAPL TSLA
```

όπου ο αριθμός 6 αντιστοιχεί στο πλήθος των εταιρειών που θέλουμε να εισάγουμε. Παρακάτω βλέπουμε τα αποτελέσματα για αυτές τις εταιρείες και $N = 1.000.000$. οι οποίες οδηγούν σε κέρδος $4,257,219,634.35816$ δολαρίων.

```
 ----> The Time Travel Project 2022 <----

Summary of results:
- Upper bound of total transactions:  1000000
- Your Companies:  {'FB', 'GE', 'TSLA', 'NFLX', 'AMZN', 'AAPL'}
- Total transactions:  17495
- Money earned:  4257219634.35816

Run time:  Hour(s):0.0, Minute(s):1.0, Seconds:18
```

και το διάγραμμα αποτίμησης που δίνεται στην τελευταία σελίδα. Τέλος, αν θέλουμε μπορούμε να εκτυπώσουμε τις κινήσεις που γίνανε στην οθόνη χρησιμοποιώντας την εντολή

```
>mainscript.py 1000000 COMPANIES 6 GE AMZN FB NFLX AAPL TSLA PRINT TRUE
```

και να δημιουργήσουμε το αρχείο txt των κινήσεων με την εντολή

```
>mainscript.py 1000000 COMPANIES 6 GE AMZN FB NFLX AAPL TSLA SAVEFILE TRUE
```

Να σημειώσουμε εδώ όταν $N <= 1000$ τότε το αρχείο αποθηκεύεται με το όνομα *small.txt* ενώ αν $N > 1000$ τότε αποθηκεύεται με το όνομα *large.txt*.

Σχήμα 3: Διάγραμμα αποτίμησης, $N = 1000000$