

ΕΛΛΗΝΙΚΟ ΜΕΣΟΓΕΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ



ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

MEMORY GAME ΣΕ ΓΛΩΣΣΑ ΡΥΘΜΩΝ

ΟΝΟΜΑ: ΒΑΣΙΛΗΣ ΚΛΙΜΑΤΣΑΚΗΣ
ΠΑΤΡΩΝΥΜΟ: ΕΜΜΑΝΟΥΗΛ ΚΛΙΜΑΤΣΑΚΗΣ
Α.Μ.: ΤΛ20270

ΕΠΙΒΛΕΠΟΝ ΚΑΘΗΓΗΤΗΣ: ΑΝΤΩΝΙΟΣ ΚΩΝΣΤΑΝΤΑΡΑΣ
ΚΡΗΤΗ, 2025

HELLENIC MEDITERRANEAN UNIVERSITY
DEPARTMENT OF ELECTRONIC ENGINEERING



MENG THESIS

MEMORY GAME WITH PYTHON PROGRAMING LANGUAGE

NAME: VASSILIS KLIMATSAKIS

FATHER'S NAME: EMMANOYIL KLIMATSAKIS

A.M.: TL20270

Supervising Professor: ANTONIOS KONSTANTARAS

Crete, 2025

Περιεχόμενα/contents

Ευχαριστίες	5
Περίληψη.....	6
Summary.....	7
1.Εισαγωγή	8
Περιγραφή της διπλωματικής.....	8
2. Python.....	10
1 Η ιστορία της python	10
2 Τι είναι η python.....	11
3 Τα οφέλη της python	12
4. Χαρακτηριστικά της python	13
5. Τρόπος μάθησης της python.....	14
6.Μελλοντικές προοπτικές της γλώσσας python.....	14
1.Data visualization.....	15
2.Data science	15
3.Artificial intelligence	15
4.Internet of things	15
3. Διερμηνευτής.....	16
1. Τι είναι διερμηνευτής	16
2.Βήματα Εκτέλεσής κώδικα σε διερμηνευτή	16
Lexical analysis.....	17
Syntax analysis.....	18
Semantic analysis.....	21
Execution	22
3 τύποι διερμηνευτών	23
Threadedcode interpreters	24
Bytecode interpreter.....	26
Abstract syntax tree interpreter	27
Stack based interpreter	28
Register based interpreter	28
4.Πλεονεκτήματα του διερμηνευτή	29

4. Παιχνίδι	30
1.Κανόνες παιχνιδιού	30
2.Τρόποι επίλυσης παιχνιδιού	31
3.Δενδροδιαγραμμα.....	32
5. Πρόγραμμα	33
1.Κώδικας.....	33
2.Λεπτομερείς ανάλυση του κώδικα.....	40
3.Εικόνες από το παιχνίδι.....	68
Συμπεράσματα	69
Πηγές	70

Ευχαριστίες:

Ευχαριστώ θερμά τον κύριο Αντώνιο Κωνσταντάρα που μου δίδαξε την python, ώστε να μπορώ να την χρησιμοποιήσω για να φτιάξω το παιχνίδι μου.

Περίληψη

Αυτό το παιχνίδι που υλοποίησα είναι ένα memory card game. Το παιχνίδι λειτουργεί ως εξής: Ο κάθε παίκτης πρέπει να επιλέξει δυο κάρτες και να δει εάν αυτές δυο κάρτες ταιριάζουν ή όχι. Αν δει ότι ταιριάζουν, τότε οι κάρτες μένουν ανοιχτές και παίρνει ένα πόντο αλλιώς αναποδογυρίζουν και ο παίκτης χάνει την σειρά του. Αυτή η διαδικασία γίνεται ώσπου ένας παίκτης ταιριάξει όλες τις κάρτες. Επιπλέον όλες οι κάρτες στην αρχή του κάθε γύρου ανακατεύονται.

Summary

This game I implemented is a **memory card game**. The game works as follows: Each player must select two cards and check if they match or not. If they match, the cards stay face-up and the player takes 1 point. Otherwise, they turn back over and the player loses his turn. This process continues until a player matches all the cards. Additionally, at the beginning of each round, all the cards are shuffled.

1.Εισαγωγή

Περιγραφή της διπλωματικής:

Σε αυτή την παρουσίαση θα μιλήσουμε για ένα παιχνίδι καθώς και θα πούμε κάποια πράγματα για την rython.

Πρώτα θα σας πω λίγα πράγματα για την rython. Αρχικά θα σας πω για την ιστορία της γλώσσας rython, τα χαρακτηριστικά της καθώς και τα οφέλη αυτής της γλώσσας. Επιπλέον θα σας πω πως μπορεί κάποιος να μάθει την rython. Επίσης θα σας πω τις μελλοντικές προοπτικές που έχει η γλώσσα rython.

Στην συνέχεια θα σας πω κάποια πράγματα σχετικά με τον κώδικα του παιχνιδιού. Αρχικά θα σας πω για την ιστορία του παιχνιδιού. Επιπλέον θα σας πω για τους κανόνες του παιχνιδιού και για τους τρόπους επίλυσης του παιχνιδιού. Στην συνέχεια θα δούμε το δένδροδιάγραμμα του παιχνιδιού. Επίσης θα σας δείξω το κώδικα που έφτιαξα για δημιουργήσω το παιχνίδι και θα σας εξηγήσω πως λειτουργεί το παιχνίδι. Μετά θα σας δείξω κάποιες εικόνες από το παιχνίδι.

Τέλος θα σας πω λίγα πράγματα για τον διερμηνευτή. Αρχικά θα σας πω τι είναι ο διερμηνευτής, τα πλεονεκτήματα του καθώς και τα βήματα εκτέλεσής ενός διερμηνευτή. Τέλος θα σας πω διάφορους τύπους διερμηνευτών.

Το παιχνίδι το έκανα στο codesculptor. Το codesculptor είναι ένας online διερμηνευτής που επιτρέπει στους χρήστες να γράφουν και να εκτελούν python προγράμματα απευθείας από το browser, χωρίς να χρειάζεται η εγκατάστασή πρόσθετου λογισμικού.

2. Python

1. Η ιστορία της python:

Η Γλώσσα python 2 [1,2,3,4,5,16,17,18] δημιουργήθηκε από τον ολλάνδο Guido Van Rossum το 1990. Η ανάπτυξη αυτής της γλώσσας ξεκίνησε το 1990 στο CWI του Άμστερνταμ και συνεχίζεται ακόμα μέχρι σήμερα στο CNRI του Reston. Η python αναπτύσσεται ως ανοιχτό λογισμικό από το οργανισμό Python software foundation.[2]

Έχουν βγει συνολικά 4 εκδόσεις. Αρχικά βγήκε η έκδοση 0.9.0, η οποία κυκλοφόρησε το Φεβρουάριο του 1991. Περιλάμβανε δυνατότητες όπως κλάσεις με κληρονομικότητα και βασικούς τύπους δεδομένων όπως λίστες και λεξικά. Πιο μετά, δηλαδή τον Οκτώβριο του 2000 βγήκε η έκδοση 2.0, η οποία περιείχε όλες τις δυνατότητές της έκδοσής 0.9.0 και κάποιες επιπρόσθετες που δεν υπήρχαν στην έκδοση 0.9.0, όπως list comprehension (κατασκευή λίστας σε μια γραμμή), διαγραφή προγραμμάτων από την μνήμη, υποστήριξη unicode. Πιο μετά, δηλαδή το Δεκέμβριο του 2008 βγήκε η έκδοση 3.0, η οποία διόρθωσε κάποια λάθη που υπήρχαν στην έκδοση 2.0, όπως στην διαίρεση ακέραιων. Η τελευταία έκδοση ήταν η 3.11, η οποία βγήκε τον Φεβρουάριο του 2023. Η τελευταία έκδοση βελτίωσε την ταχύτητα εκτέλεσής κώδικα.[1]

2. Τι είναι η python:



Εικόνα 1: Λογότυπο της python [16]

Η Γλώσσα python είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου [2,16], η οποία έχει σαν στόχο να κάνει το προγραμματιστή να κάνει πολύπλοκα πράγματα παρά πολύ εύκολα και ο κώδικας να διαβάζεται εύκολα από κάποιον τρίτο, εκτός από αυτόν που έγραψε τον κώδικα.

3. Τα οφέλη της python:

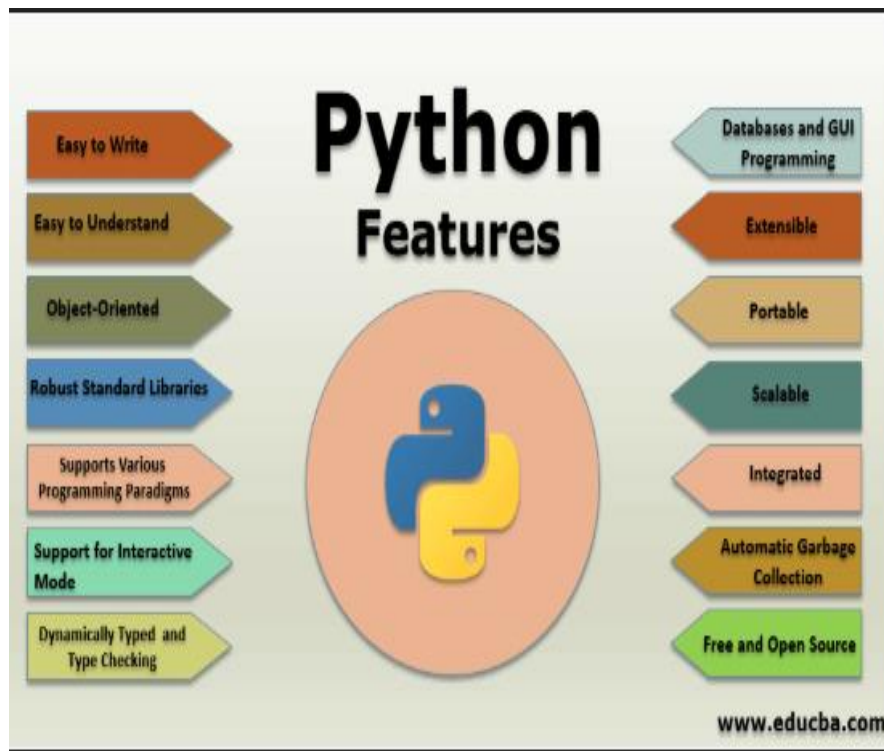
Η python έχει πολλά οφέλη [3,17]. Κάποια από τα οφέλη της είναι:

1. Είναι μια απλή γλώσσα προγραμματισμού
2. Είναι προσανατολισμένα στην δημιουργία μικρότερων σε έκταση προγραμμάτων
3. Εύκολη αναγνώριση λαθών στο κώδικα.
4. Είναι εύκολη η εκμάθηση της γλώσσας.



Εικόνα 2: Πλεονεκτήματα της python [17]

4. Χαρακτηριστικά της python:



Εικόνα 3: Τα χαρακτηριστικά της python [18]

Τα χαρακτηριστικά της python παρουσιάζονται παρακάτω [4,18]:

1. Έχει απλή σύνταξη
2. Είναι μια γλώσσα προγραμματισμού που χρησιμοποιεί διερμηνευτή
3. Μπορείς να ενσωματωθεί σε άλλες γλώσσες προγραμματισμού

5. Τρόπος μάθησης της python:

Υπάρχουν οι εξής τρόποι για να μάθει κάποιος την γλώσσα python:

1. Μέσω ενός βιβλίου
2. Μεσω του documentation του codesculptor
3. Μέσω βίντεο στο youtube
4. Από διάφορες πηγές στο διαδίκτυο
5. Μέσω των έτοιμων demo που υπάρχουν στο codesculptor

6.Μελλοντικές προοπτικές της γλώσσας python:

Η python μπορεί να βοηθήσει σε πολλούς τομείς [10],όπως:

1. Data visualization
2. Data science
3. Artificial intelligence
4. Internet of things

1.Data visualization

Η python έχει αρκετές βιβλιοθήκες για την οπτικοποίηση των δεδομένων,όπως Matplotlib and plotly,τα οποία μπορεί να χρησιμοποιήσει ένας development για να δημιουργήσει όμορφα visuals για να αποικωνήσει όμορφα σημαντικά συμπεράσματα η δεδομένα.

2.Data science

Η Python είναι μια γλώσσα προγραμματισμού,η οποία έχει μια μεγάλη γκάμα από βιβλιοθήκες για επεξεργασία και ανάλυση δεδομένων.Κάποιες από αυτές τις βιβλιοθήκες είναι οι Pandas και NumPy,οι οποίες παρέχουν χρήσιμα εργαλεία για την διαχείριση μεγάλων συλλογών δεδομένων,την στατιστική ανάλυση καθώς και την κατασκευή προγνωστικών μοντέλων.

3.Artificial intelligence

Η python έχει πολλές βιβλιοθήκες για την δημιουργία και την εκπαίδευση αι μέσω διάφορων αλγορίθμων εκπαίδευσής. Μερικές από αυτές τις βιβλιοθήκες είναι οι Tensorflow, PyTorch και scikit-learn.

4.Internet of things

Η python έχει βιβλιοθήκες για τον απομακρυσμένο έλεγχο διάφορων πραγμάτων μέσω μιας πλατφόρμας στο internet.Ένα παράδειγμα μια τέτοιας εφαρμογής είναι το smart home.Μερικές από αυτές τις βιβλιοθήκες είναι MicroPython and CircuitPython .

3. Διερμηνευτής

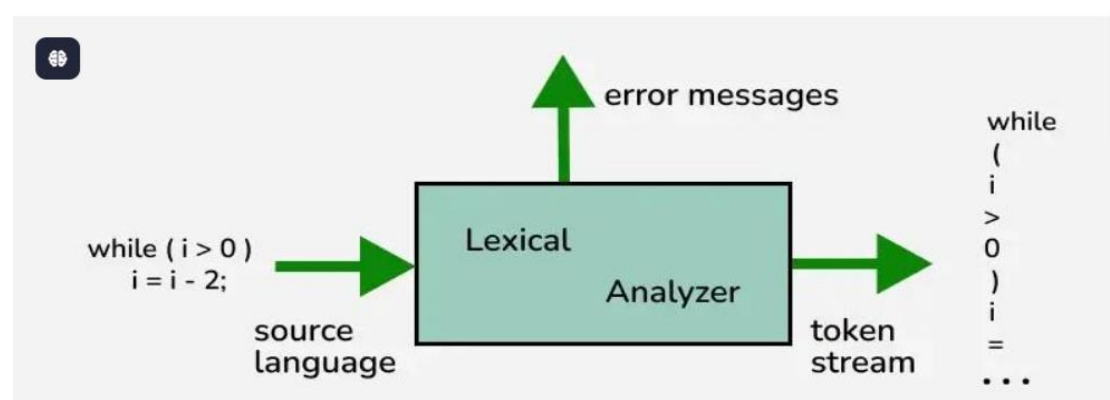
1. Τι είναι διερμηνευτής:[7]

Η python είναι γλώσσα διερμηνευτή. [6,7,8,9,10,11,12,13,14,15,20]. Διερμηνευτής(interpreter) είναι ένα πρόγραμμα, το οποίο εκτελεί απευθείας τον πηγαίο κώδικα γραμμή προς γραμμή χωρίς να χρειάζεται να μετατραπεί σε εκτελέσιμο αρχείο πρώτα.

2. Βήματα Εκτέλεσής κώδικα σε διερμηνευτή:

Τα στάδια μέσα από τα οποία περνάει ένα πρόγραμμα σε γλώσσα python [6,8,9,10,11,12,13,20] είναι lexical analysis, syntax analysis, semantic analysis και execution.

Lexical analysis



Εικόνα 4: Δομή lexical analyzer [8]

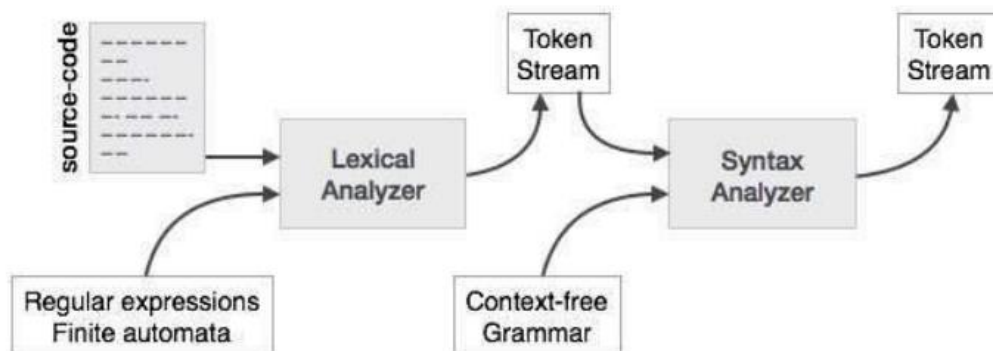
Σε αυτή την φάση [6,8,9] ο λεξικός αναλυτής(lexer)

1. Διαβάζει τον πηγαίο κώδικα χαρακτήρα προς χαρακτήρα.
2. Ομαδοποιεί αυτούς τους χαρακτήρες σε tokens βάση κανόνων που έχουν προκαθοριστεί από την python.
3. Αφαιρούνται τα κενά διαστήματα και τα σχόλια.
4. Ελέγχει για λεξικά σφάλματα(πχ σύμβολα που δεν αναγνωρίζει η γλώσσα ή αναγνωριστικά,τα οποία δεν τηρούν τους βασικούς κανόνες που πρέπει να εφαρμόζει ένα αναγνωριστικό για να είναι σωστό).

Τα tokens μπορεί να είναι:

- Είτε λέξεις κλειδιά(πχ if,while,return)
- Αναγνωριστικά(ονόματα μεταβλητών και συναρτήσεων)
- Τελεστές
- Σημεία στίξης(πχ :)

Syntax analysis



Εικόνα 5:lexical and syntax analyzer[20]

Σε αυτή την φάση [10,11,12,20] ο parser παίρνει τα tokens από τον lexical analyser και τα οργανώνει σε μια ιεραρχική δομή(parse tree or syntax tree). Τις περισσότερες φορές αυτή η δομή είναι ένα δέντρο σύνταξης(syntax tree).Δηλαδή ο parser κοιτάζει τα syntax trees και ελέγχει για συντακτικά λάθη(πχ.εάν στο τέλος μιας συνθήκης if υπάρχει η άνω κάτω τελεία).

Syntax tree:

Το syntax tree (ή δέντρο σύνταξης) [11] είναι μια δενδρική δομή που δείχνει πώς συνδέονται και οργανώνονται τα tokens μεταξύ τους σύμφωνα με τους κανόνες της γραμματικής της γλώσσας προγραμματισμού.

Production rules είναι κανόνες που λένε πως ένα τερματικό σύμβολο μπορεί να μετατραπεί σε μια σειρά από τερματικά ή

μη τερματικά σύμβολα. Τα production rules εφαρμόζονται μέχρι όλη η έκφραση να έχει μόνο τερματικά σύμβολα.

Μη-Τερματικά σύμβολα είναι σύμβολα που μπορούν να αντικατασταθούν χρησιμοποιώντας τους κανόνες παραγωγής.

Τερματικά σύμβολα είναι σύμβολα που δεν μπορούν να αντικατασταθούν παραπέρα.

Έστω ότι έχουμε το string $A \rightarrow x * y + B$. Στα αριστερά αυτού του string το A είναι τερματικό ενώ από τα δεξιά αυτού του string το x, y, * είναι τερματικά σύμβολα και το B είναι μη τερματικό.

Έχουμε δυο τρόπους με τους εφαρμόζουμε τους production rules:

- Right-most Derivation

Σε αυτήν την περίπτωση αντικαθιστούμε τους production rules σε ένα input string από δεξιά προς αριστερά.

- Left-most Derivation

Σε αυτήν την περίπτωση αντικαθιστούμε τους production rules σε ένα input string από αριστερά προς δεξιά.

Έχουμε πχ το input string: $F = 3 * 4 + 5$

Στην περίπτωση που χρησιμοποιήσουμε Right-most Derivation τότε το string θα αλλάξει ως εξής:

$$E \longrightarrow E+E$$

$$E \longrightarrow E+id$$

$$E \longrightarrow E*E+id$$

$$E \longrightarrow E*id+id$$

$$E \longrightarrow id*id+id$$

Στην περίπτωση που χρησιμοποιήσουμε left-most Derivation τότε το string θα αλλάξει ως εξής:

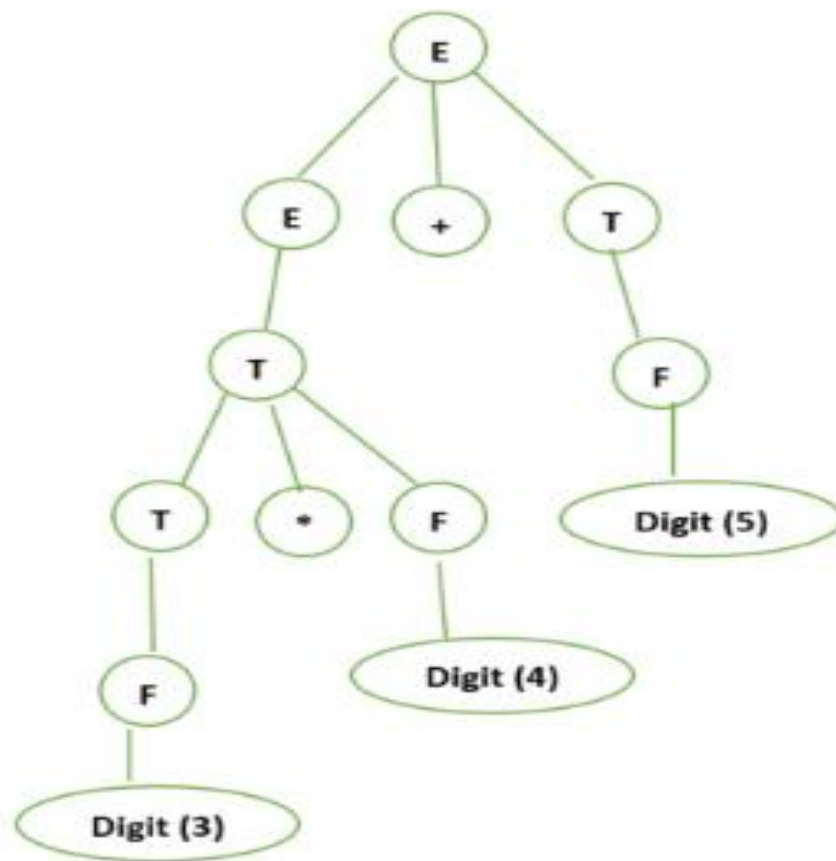
$$E \longrightarrow E+E$$

$$E \longrightarrow E*E+E$$

$$E \longrightarrow id*E+E$$

$$E \longrightarrow id*id+E$$

$$E \longrightarrow id*id+id$$



Semantic analysis

Σε αυτό το στάδιο κοιτάμε το πρόγραμμα για σημασιολογικά λάθη [6,13].

Οι εργασίες που γίνονται στην σημασιολογική ανάλυση είναι:

1. κατανόηση της κάθε λέξης ενός input string με βάση τα συμφραζόμενα
2. Η κατανόηση των σχέσεων που υπάρχουν μεταξύ των λέξεων ενός input string.

Πχ. Εάν έχουμε το input string:: $F=3*4+5$, τότε ο σημασιολογικός αναλυτής ελέγχει:

3. Τι τύπο έχει η κάθε έκφραση(δηλαδή ελέγχει πρώτα το $3*4$ και βλέπει ότι και οι αριθμοί 3 και 4 έχουν τύπο int και πάει μετά στο 5 και βλέπει ότι και αυτός ο αριθμός είναι τύπου int)
4. Εάν οι τελεστές εφαρμόζονται στους σωστούς τύπους δεδομένων(δηλαδή βλέπει ότι οι αριθμοί 3 και 4, στους οποίους εφαρμόζεται ο τελεστής * είναι σωστού τύπου(όποτε κάνει κανονικά τον πολλαπλασιασμό. Μετά παίρνει το $3*4$ σαν έκφραση, η οποία έχει τύπο int και βλέπει ότι η έκφραση $3*4$ και ο αριθμός 5 έχουν και τα δυο τύπο int(όποτε κάνει την πρόσθεση κανονικά).

Execution

Σε αυτό το στάδιο [6] ο interpreter εκτελεί το πρόγραμμα απευθείας από την ενδιάμεση μορφή ή από τα syntax trees .

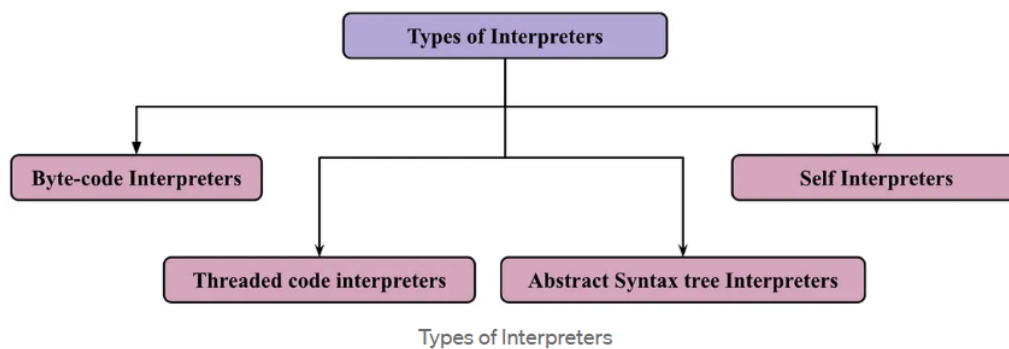
Αν υπάρξει κάποιο σφάλμα κατά την εκτέλεση του προγράμματος, τότε ο interpreter σταματάει την εκτέλεση και εμφανίζει μήνυμα λάθους σταματώντας την εκτέλεση του προγράμματος.

3. Τύποι διερμηνευτών:

Υπάρχουν διάφοροι τύποι διερμηνευτών [14,15]. Μερικούς από αυτούς είναι:

- Threaded code interpreters
- Abstract syntax tree interpreter
- Bytecode interpreter
- Stack based interpreter
- Register Based interpreter

TYPES OF INTERPRETERS



Εικόνα 6:τύποι διερμηνευτών [15]

Threaded-code interpreters:

Οι διερμηνευτές αυτοί διαχειρίζονται έναν πίνακα από δείκτες, όπου κάθε δείκτης αντιστοιχεί σε μια εντολή του προγράμματος. Κάθε γραμμή του πίνακα περιέχει:

- την ίδια την εντολή (ή το όνομα μιας συνάρτησης)?
- τις αντίστοιχες παραμέτρους της, διαχωρισμένες με κομματάκι ώστε να μην υπάρχει σύγχυση.

Η εκτέλεση γίνεται σειριακά: ξεκινώντας από την πρώτη γραμμή του πίνακα, κάθε δείκτης ενεργοποιεί την εντολή στην οποία δείχνει. Μετά την εκτέλεση, η διαδικασία συνεχίζεται με την επόμενη γραμμή, μέχρι να ολοκληρωθούν όλες οι εντολές.

Ενας κωδικας threadedcode interpreter είναι:

```
program = [  
    ("print", "Hello!"),  
    ("add", 10, 20),  
    ("multiply", 3, 4),  
]
```

for instruction in program:

```
    cmd = instruction[0]
```

```
if cmd == "print":  
    print(instruction[1])  
  
elif cmd == "add":  
    a, b = instruction[1], instruction[2]  
    print(f"{a} + {b} = {a + b}")  
  
elif cmd == "multiply":  
    a, b = instruction[1], instruction[2]  
    print(f"{a} × {b} = {a * b}")
```

Bytecode interpreter:

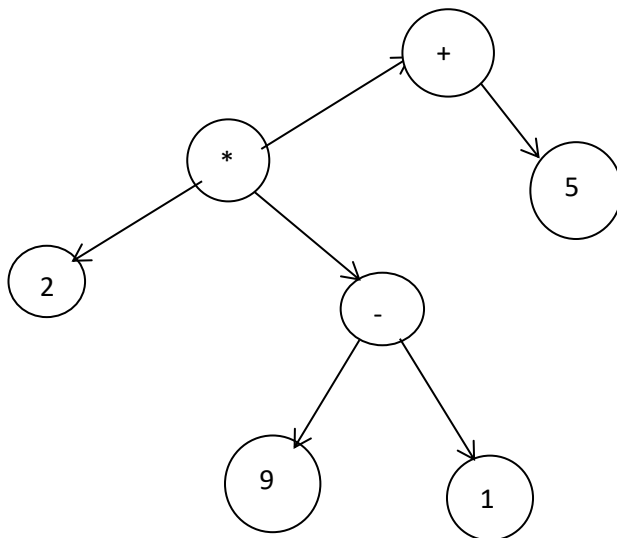
Οι διερμηνευτές αυτοί παίρνουν το syntax tree και το μετατρέπουν σε εντολές bytecode και μετά τις τρέχει μια-μια εκτελώντας την λειτουργία που αντιστοιχεί σε κάθε bytecode. Ο bytecode είναι ένας πίνακας από οδηγίες, όπου κάθε οδηγία αποτελείται από:

- Το opcode, το οποίο δηλώνει ποια λειτουργία πρόκειται να εκτελεστεί.
- επιπλέον πληροφορίες που χρειάζονται για να εκτελεστεί η εντολή.

Abstract syntax tree interpreter:

Οι διερμηνευτές αυτοί παίρνουν τον κώδικα γραμμή-γραμμή και φτιάχνουν ένα δέντρο AST, όπου αντικατοπτρίζει την λογική του προγράμματος. Μετά ο διερμηνευτής εκτελεί το πρόγραμμα με βάση το AST.

Έστω ότι έχουμε την αριθμητική πράξη $2*(9-1)+5$. Τότε το AST θα γίνει:



Το AST προσπερνιέται από αριστερά προς δεξιά.

Stack based interpreter:

Οι διερμηνευτές (interpreters) διαβάζουν τον πηγαίο κώδικα γραμμή-γραμμή, αναλύουν την κάθε γραμμή, εκτελούν τις εντολές (όπως πράξεις με αριθμούς) και αποθηκεύουν τα ενδιάμεσα ή τελικά αποτελέσματα, σε μια στοίβα.

Register based interpreter:

Αυτοί οι διερμηνευτές είναι όπως τους stack based interpreters με την διαφορά ότι αντί να αποθηκεύουν τους τελεστές και τα ενδιάμεσα αποτελέσματα σε στοίβα τα αποθηκεύουν σε εικονικούς καταχωρητές (virtual registers).

Έστω ότι έχουμε την αριθμητική πράξη $(2*3)+5$. Τότε οι εντολές του register based interpreter θα είναι:

LOAD R1,3

LOAD R2,2

LOAD R3,5

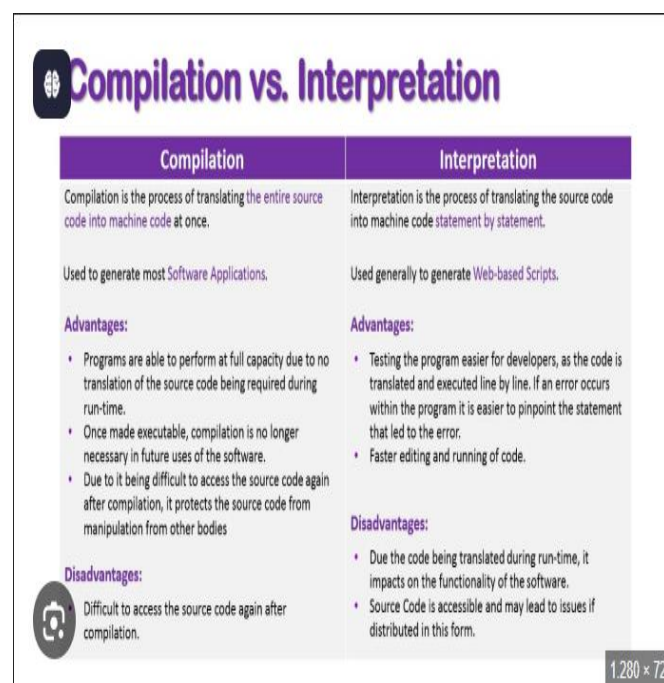
MUL R4,R1,R2

ADD R5,R4,R3

4.Πλεονεκτήματα του διερμηνευτή:

Τα πλεονεκτήματα του διερμηνευτή [19] είναι:

1. Οι προγραμματιστές μπορούν να κάνουν εκτέλεση μια-μια την γραμμή και να δούμε τα αποτελέσματα κατευθείαν.
2. Έχουν πιο φιλικό περιβάλλον προς τον χρήστη.
3. Μπορεί να τρέξει σε οποιαδήποτε πλατφόρμα, η οποία έχει interpreter, ανεξαρτήτως του λειτουργικού συστήματος.
4. Βοηθάει στους προγραμματιστές να γράψουν πιο γρήγορα κώδικα, αφού βλέπουν κατευθείαν το αποτέλεσμα.
5. Βοηθούν τον προγραμματιστή στο να βρει πιο γρήγορα το λάθος και να το διορθώσει, αφού έχει πιο αναλυτικά error messages.



Compilation	Interpretation
Compilation is the process of translating the entire source code into machine code at once.	Interpretation is the process of translating the source code into machine code statement by statement.
Used to generate most Software Applications.	Used generally to generate Web-based Scripts.
Advantages: • Programs are able to perform at full capacity due to no translation of the source code being required during run-time. • Once made executable, compilation is no longer necessary in future uses of the software. • Due to it being difficult to access the source code again after compilation, it protects the source code from manipulation from other bodies	Advantages: • Testing the program easier for developers, as the code is translated and executed line by line. If an error occurs within the program it is easier to pinpoint the statement that led to the error. • Faster editing and running of code.
Disadvantages: • Difficult to access the source code again after compilation.	Disadvantages: • Due the code being translated during run-time, it impacts on the functionality of the software. • Source Code is accessible and may lead to issues if distributed in this form.

Εικόνα 7: compiler advantages[19]

4. Παιχνίδι

1.Κανόνες παιχνιδιού:

Ανάστροφη καρτών:

- Σε κάθε γύρο ένας παίκτης πρέπει να γυρίσει δυο κάρτες
- Αν ταιριάζουν μεταξύ τους, οι κάρτες μένουν ανοιχτές
- Αν δεν ταιριάζουν οι κάρτες κλείνουν

Εναλλαγή σειράς:

- Αν οι κάρτες ταιριάζουν, τότε ο ίδιος παίκτης ξαναπαίζει
- Αν οι κάρτες δεν ταιριάζουν, ο παίκτης χάνει την σειρά του

Σκορ:

- Όποιος παίκτης βρει ένα ζευγάρι κερδίζει ένα πόντο

Τέλος παιχνιδιού:

- Ο πρώτος παίκτης που θα βρει όλα τα ζευγάρια κερδίζει

Κουμπιά:

New game:

Ξεκινά νέο παιχνίδι με ανακατεμένες κάρτες και η σειρά πάει αυτόματα στον πρώτο παίκτη.

Reset:

Μηδενίζει το σκορ του κάθε παίκτη κλείνοντας όλες τις ανοιχτές κάρτες και η σειρά πάει αυτόματα στον πρώτο παίκτη.

2.Τρόποι επίλυσης παιχνιδιού:

Υπάρχουν διάφοροι τρόποι για να καταφέρει κάποιος να φτάσει στην νίκη σε αυτό το παιχνίδι. Αυτοί είναι:

1. Μνήμη και παρατήρηση:

Μπορείς να θυμάσαι την θέση των καρτών που έχεις ανοίξει, ώστε να μπορέσεις να κάνεις ποιο καλύτερες επιλογές ποιο μετά στο παιχνίδι.

2. Παρακολούθηση του αντιπάλου:

Σε περίπτωση που δεν βρει το ζευγάρι ο αντίπαλος, μπορείς να θυμάσαι ποιες κάρτες άνοιξε, ώστε όταν έρθει η σειρά σου να έχεις μεγαλύτερες πιθανότητες να βρεις το ζευγάρι που έψαχνες.

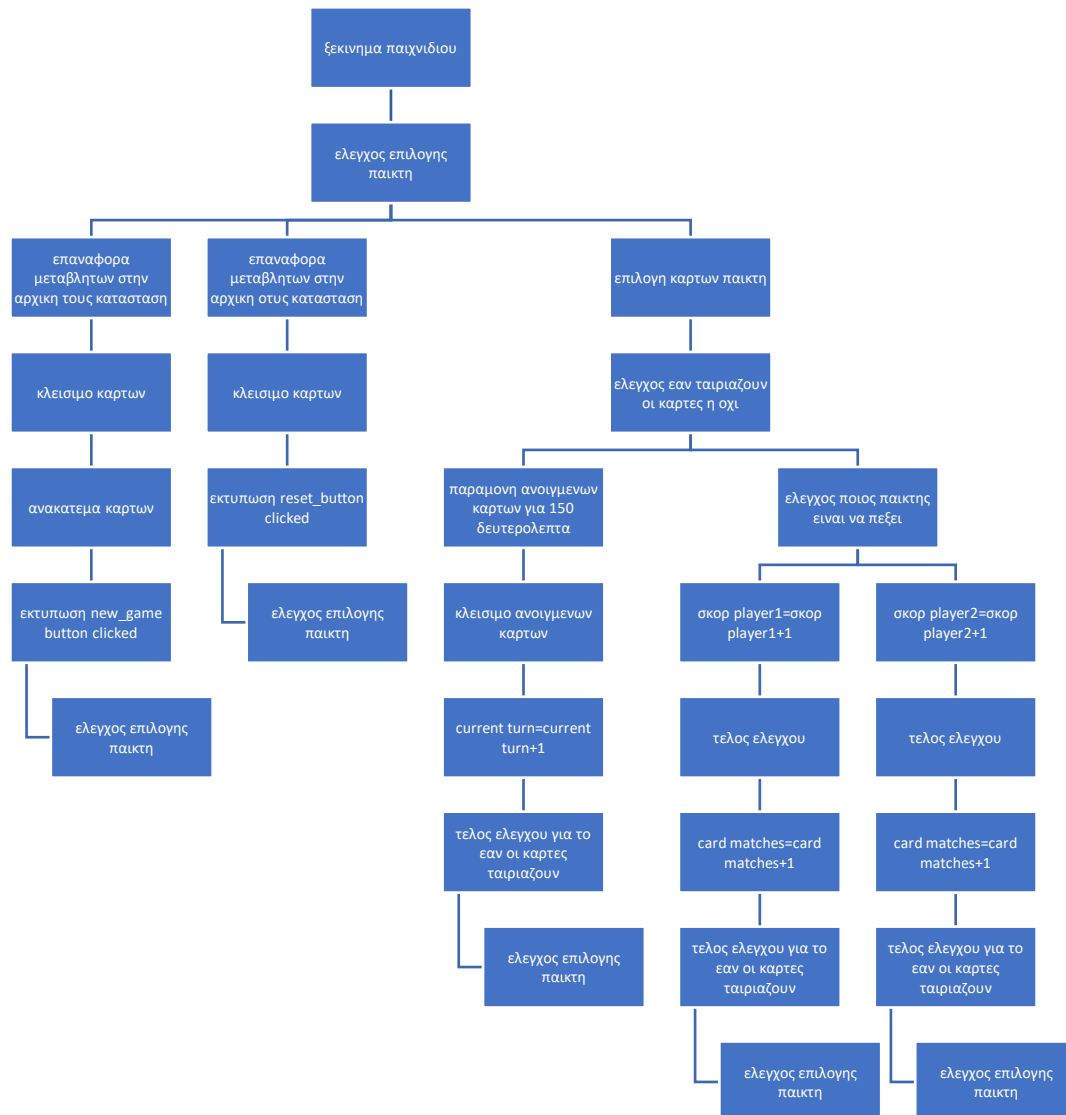
3. Εξερεύνηση των θέσεων των συμβόλων:

Αντί να προσπαθείς να βρεις τα ζευγάρια στην αρχή του παιχνιδιού μπορείς να ανοίξεις όσες περισσότερες κάρτες μπορείς, για να μαζέψεις πληροφορίες.

4. Απομνημόνευση όλων των καρτών που έχουν ανοίξει:

Μπορείς να θυμάσαι όλες τις κάρτες που έχετε ανοίξει εσύ και ο αντίπαλος σου, άμα έχεις πολύ καλή μνήμη.

3.Δενδροδιαγραμμα:



Κάτω από το κουτάκι που λέει έλεγχος επιλογής παίκτη έχουμε 3 παρακλάδια, ένα κουτάκι στο κέντρο που είναι στην περίπτωση που ο παίκτης πατήσει το reset button, ένα που είναι αριστερά που είναι στην περίπτωση που παίκτης πατήσει το new game button, και ένα δεξιά που είναι στην περίπτωση που ο παίκτης επιλέξει μια δυάδα καρτών.

5. Πρόγραμμα

1.Κώδικας

```
import pygame
import random
import time

pygame.init()

# Ρυθμίσεις παραθύρου
SCREEN_SIZE = 800
CARD_SIZE = 100
GRID_SIZE = 4
GAP=120
BACKGROUND_COLOR = (255, 255, 255)

# Δημιουργία παραθύρου
screen = pygame.display.set_mode((SCREEN_SIZE*1.7, SCREEN_SIZE))
pygame.display.set_caption("Memory Card Game")

# Συνάρτηση ανακατέματος καρτών
def shuffle_cards():
    symbols = ['A', 'A', 'B', 'B', 'C', 'C', 'D', 'D', 'E', 'E', 'H', 'H', 'K', 'K', 'F', 'F']
    random.shuffle(symbols)
    return [symbols[i * GRID_SIZE:(i + 1) * GRID_SIZE] for i in range(GRID_SIZE)]
```

```
# Δημιουργία πλέγματος καρτών
```

```
cards = shuffle_cards()
```

```
revealed = [[False] * GRID_SIZE for _ in range(GRID_SIZE)]
```

```
def reset():
```

```
    global score_player_1, score_player_2, card_matches, current_turn, revealed
```

```
    score_player_1=0
```

```
    score_player_2=0
```

```
    current_turn=0
```

```
    card_matches=0
```

```
    revealed = [[False] * GRID_SIZE for _ in range(GRID_SIZE)]
```

```
    print("reset button clicked!")
```

```
def new_game():
```

```
    global score_player_1, score_player_2, card_matches, current_turn, revealed, cards
```

```
    score_player_1=0
```

```
    score_player_2=0
```

```
    current_turn=0
```

```
    card_matches=0
```

```
    revealed = [[False] * GRID_SIZE for _ in range(GRID_SIZE)]
```

```
    cards=shuffle_cards()
```

```
    print("new_game button clicked!")
```

```
def get_card_at_pos(pos):
```

```
    x, y = pos
```

```
    for row in range(GRID_SIZE):
```

```
        for col in range(GRID_SIZE):
```

```
            card_x = col * (CARD_SIZE + GAP)
```

```

        card_y = row * (CARD_SIZE + GAP)

        if card_x <= x <= card_x + CARD_SIZE and card_y <= y <= card_y + 2 * CARD_SIZE:
            return row, col

    return None

reset_rect=pygame.Rect(900,350,90,40)
new_game_rect=pygame.Rect(1050,350,160,40)


running = True
reveal_time = None
first_selection = None
second_selection = None
score_player_1=0
score_player_2=0
current_turn=0
card_matches=0


while running:

    screen.fill(BACKGROUND_COLOR) # Καθαρισμός οθόνης


    for ev in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

        elif event.type == pygame.MOUSEBUTTONDOWN:

```

```

pos = pygame.mouse.get_pos()
card_pos = get_card_at_pos(pos)
if reset_rect.collidepoint(pos):
    reset()
elif new_game_rect.collidepoint(pos):
    new_game()
elif card_pos and not revealed[card_pos[0]][card_pos[1]]:
    if first_selection is None:
        first_selection = card_pos
        revealed[card_pos[0]][card_pos[1]] = True

    elif second_selection is None:
        second_selection = card_pos
        revealed[card_pos[0]][card_pos[1]] = True
        reveal_time = pygame.time.get_ticks()

if reveal_time and pygame.time.get_ticks() - reveal_time > 150:
    if first_selection and second_selection:
        r1,c1=first_selection
        r2,c2=second_selection
        if cards[r1][c1]==cards[r2][c2]:
            if current_turn%2==0:
                score_player_1=score_player_1+1
                revealed[r1][c1]=True
                revealed[r2][c2]=True
                card_matches=card_matches+1

            elif current_turn%2==1:
                score_player_2=score_player_2+1
                revealed[r1][c1]=True
                revealed[r2][c2]=True

```

```

        card_matches=card_matches+1
    else:
        pygame.time.delay(150)
        current_turn =current_turn+1
        revealed[r1][c1]=False
        revealed[r2][c2]=False
        first_selection=None
        second_selection=None

# Σχεδίαση καρτών
for row in range(GRID_SIZE):
    for col in range(GRID_SIZE):
        x = col * (CARD_SIZE + GAP)
        y = row * (CARD_SIZE + GAP)
        if revealed[row][col]:
            pygame.draw.rect(screen, (0, 0, 0), (x, y, CARD_SIZE, CARD_SIZE * 2), 5)
            font = pygame.font.Font('freesansbold.ttf', 32)
            text = font.render(cards[row][col], False, (255, 0, 0))
            positions = [
                (x + 15, y + 15), # Πάνω αριστερά
                (x + CARD_SIZE - 15, y + 15), # Πάνω δεξιά

                (x + CARD_SIZE // 2 - 15, y + CARD_SIZE // 2+15),
                (x + CARD_SIZE // 2+15, y + CARD_SIZE // 2+15 ),
                (x + CARD_SIZE // 2 - 15, y + CARD_SIZE // 2+45),
                (x + CARD_SIZE // 2 +15, y + CARD_SIZE // 2+45),
                (x + CARD_SIZE // 2 - 15, y + CARD_SIZE // 2+75),
                (x + CARD_SIZE // 2 + 15, y + CARD_SIZE // 2+75),
                (x + 15, y + 175),

```

```

(x + CARD_SIZE-15, y + 175),

# Κέντρο πάνω

# Κέντρο αριστερά

# Κέντρο δεξιά

# Κάτω δεξιά

]

# Σχεδίαση των συμβόλων στις σωστές θέσεις

for pos in positions:

    text_rect = text.get_rect(center=pos)

    screen.blit(text, text_rect)

else:

    # Σχεδίαση κρυφής κάρτας (κόκκινο ορθογώνιο)

    pygame.draw.rect(screen, (255, 0, 0), (x, y, CARD_SIZE, CARD_SIZE * 2), 0)

pygame.draw.rect(screen, (150, 0, 0), (780, 50, 500, 100), 0)

font = pygame.font.Font('freesansbold.ttf', 32)

texts1 = font.render('score player 1', False, (255, 0, 0))

text_rects1 = texts1.get_rect(center=(900, 70))

screen.blit(texts1, text_rects1)

pygame.draw.line(screen, (0, 0, 0), (1025, 50), (1025, 150), 5)

texts2 = font.render('score player 2', False, (255, 0, 0))

text_rects2 = texts2.get_rect(center=(1155, 70))

screen.blit(texts2, text_rects2)

pygame.draw.rect(screen, (255, 255, 255), (800, 80, 205, 60), 0)

pygame.draw.rect(screen, (255, 255, 255), (1050, 80, 205, 60), 0)

textns1 = font.render(str(score_player_1), False, (255, 0, 0))

text_rectns1 = textns1.get_rect(center=(900, 110))

screen.blit(textns1, text_rectns1)

textns2 = font.render(str(score_player_2), False, (255, 0, 0))

```

```
text_rectns2 = textns2.get_rect(center=(1155,110))
screen.blit(textns2, text_rectns2)
turn_text = font.render(f"{current_turn % 2 + 1}'s Turn", False, (0, 153, 0))
turn_text_rect = turn_text.get_rect(center=(1030, 250))
screen.blit(turn_text, turn_text_rect)
pygame.draw.rect(screen,(96,96,96),reset_rect)
textbure = font.render('reset', False, (255, 0, 0))
textbure_rect = textbure.get_rect(center=(940,370))
screen.blit(textbure, textbure_rect)
pygame.draw.rect(screen,(96,96,96),new_game_rect)
textbung = font.render('new game', False, (255, 0, 0))
text_rectbung = textbung.get_rect(center=(1130,370))
screen.blit(textbung, text_rectbung)
pygame.display.update()
```

2.Λεπτομερείς ανάλυση του κώδικα:

*/*Με το import εξάγουμε έτοιμες βιβλιοθήκες από την pygame.Εδώ πέρα κάνουμε εξαγωγή της έτοιμης βιβλιοθήκης που θα χρειαστούμε για τα γραφικά του παιχνιδιού, δηλαδή την βιβλιοθήκη pygame.*/*

```
import pygame
```

*/*Κάνουμε εξαγωγή της βιβλιοθήκης random για τις τυχαίες συναρτήσεις που θα χρειαστούμε για το ανακάτεμα των καρτών.*/*

```
import random
```

*/*Κάνουμε εξαγωγή της βιβλιοθήκης time για το μέτρημα του χρόνου που θα μείνουν ανοιχτές οι κάρτες για τα ζευγάρια που δεν ταιριάζουν μεταξύ τους.*/*

```
import time
```

*/*Εδώ πέρα με το init κάνουμε εκκίνηση της βιβλιοθήκης pygame.*/*

```
pygame.init()
```

*/*Εδώ πέρα έχουμε μια μεταβλητή που έχουμε ονομάσει SCREEN_SIZE και της έχουμε δώσει τιμή 800,η οποία θα είναι το μέγεθος του παραθύρου της pygame.*/*

```
SCREEN_SIZE = 800
```

```
/*Φτιάχνουμε μια μεταβλητή με όνομα CARD_SIZE ,για το μέγεθος της κάρτας.*/
```

```
CARD_SIZE = 100
```

```
/*Φτιάχνουμε μια μεταβλητή με το όνομα GRID_SIZE,της οποίας έχουμε θέσει κάποια τιμή που λέει πόσες κάρτες θα έχει η κάθε γραμμή και η κάθε στήλη.*/
```

```
GRID_SIZE = 4
```

```
/*Φτιάχνουμε μια μεταβλητή που λέει πόσο θα είναι το κενό μεταξύ των καρτών.*/
```

```
GAP=120
```

```
/*Φτιάχνουμε μια μεταβλητή με τιμή rgb color(255,255,255).Το (255,255,255) είναι άσπρο. Αυτό το rgb color θα είναι το background color του παραθύρου pygame.*/
```

```
BACKGROUND_COLOR = (255, 255, 255)
```

```
/*Δημιουργούμε το παράθυρο pygame με τις κατάλληλες διαστάσεις στο άξονα x και y.Στην συγκεκριμένη περίπτωση ο άξονας x είναι ίσος με (μέγεθος του παραθύρου pygame)*1,7 και ο άξονας y είναι ίσος με μέγεθος παραθύρου. Στο pygame.display.set_mode το display είναι modules μέσα στην βιβλιοθήκη pygame και το set_mode είναι συνάρτηση μέσα στο module display.Με το display δημιουργούμε το παράθυρο pygame και με το set_mode ορίζουμε τις διαστάσεις του παραθύρου.*/
```

```
pygame.display.set_mode((SCREEN_SIZE*1.7, SCREEN_SIZE))
```

```
/*Εδώ πέρα έχουμε πάλι το module display της βιβλιοθήκης  
pygame και την συνάρτηση set_caption του module display. Με  
την συνάρτηση set_caption βάζουμε τίτλο στο παράθυρο  
pygame.*/
```

```
pygame.display.set_caption("Memory Card Game")
```

```
/*Κάνουμε ορισμό της συνάρτησης shuffle_cards(),η οποία θα  
ανακατεύει κάθε φορά τις κάρτες, έτσι ώστε να υπάρχει κάποια  
τυχειότητα στην τοποθέτηση των συμβολών στις κάρτες κάθε  
φορά. Η συνάρτηση αυτή, όπως βλέπουμε δεν παίρνει κάποια  
είσοδο.*/
```

```
def shuffle_cards():
```

```
/*Δημιουργούμε μια λίστα με τα σύμβολα της κάρτας. Μια  
λίστα ξεκινάει πάντα από το μηδέν και φτάνει μέχρι το αριθμό  
των στοιχείων που υπάρχουν στην λιστα-1*/
```

```
symbols = ['A', 'A', 'B', 'B', 'C', 'C', 'D', 'D', 'E', 'E', 'H', 'H', 'K', 'K', 'F',  
'F']
```

```
/*Εδώ πέρα ανακατεύουμε τυχαία τα σύμβολα της δομής  
symbols.Έχουμε την συνάρτηση shuffle της βιβλιοθήκης  
random,η οποία παίρνει σαν είσοδο(παράμετρο) την  
ακολουθία που είναι να ανακατέψει*/
```

```
random.shuffle(symbols)
```

```
/*Εδώ πέρα ορίζουμε τι θα επιστρέψει η συνάρτηση  
shaffle_cards().Αυτό που θα επιστρέψει η συνάρτηση shuffle  
cards θα είναι μια λίστα με 4 υπολίστες. Η ανακατεμένη λίστα
```

χωρίζεται σε υποτίτλισες των 4 στοιχείων η καθεμία. Το i ξεκινάει από το 0 και φτάνει μέχρι $GRID_SIZE-1$. Για κάθε i δημιουργείται μια υπολίστα από την θέση $i*GRID_SIZE$ μέχρι $(i+1)*GRID_SIZE$. Δηλαδή η πρώτη υπολίστα θα έχει τα ανακατεμένα σύμβολα από την θέση $0*4=0$ μέχρι(;) την θέση $(0+1)*4=1*4$ από την ανακατεμένη λίστα που μας γύρισε η συνάρτηση `shuffle`, η δεύτερη υπολίστα από την θέση $1*4=4$ μέχρι την θέση $(1+1)*4=2*4=8$, η τρίτη υπολίστα από την θέση 8 έως την θέση 12 και η τέταρτη υπολίστα από την θέση 12 έως την θέση 16.*/

```
return [symbols[i * GRID_SIZE:(i + 1) * GRID_SIZE] for i in
range(GRID_SIZE)]
```

/*Εδώ πέρα βάζουμε την λίστα που θα μας επιστρέψει η συνάρτηση `shuffle` σε μια μεταβλητή(`cards`).*/

```
cards = shuffle_cards()
```

/*Εξωτερικός βρόγχος(`for _ in range(GRID_SIZE)`):

Εκτελείται 4 φορές όσες δηλαδή είναι το `grid_size`. Για κάθε επανάληψη δημιουργεί μια λίστα με 4 τιμές `False`. Το `_` χρησιμοποιείται επειδή δεν μας ενδιαφέρει η τιμή του μετρητή.

Εσωτερική λίστα(`[False] * GRID_SIZE`):

Είναι αυτό που εκτελείται σε κάθε επανάληψη.

Το αποτέλεσμα είναι η δημιουργία μιας λίστας με 4 υπολίστες `[False, False, False, False]`.*/

```
revealed = [[False] * GRID_SIZE for _ in range(GRID_SIZE)]
```

```
/*Ορίζουμε την συνάρτηση reset(), η οποία κάνει reset του  
ταμπλό του παιχνιδιού στην αρχική του κατάσταση.*/
```

```
def reset():
```

```
/*Ορίζουμε μεταβλητές τύπου global για το σκορ του 1 παίκτη,  
για το σκορ του δεύτερου παίκτη ,για τα ζευγάρια,για την σειρά  
του παίκτη και για τις ανοιχτές κάρτες.Αυτό το κάνουμε για να  
μπορούμε να χρησιμοποιήσουμε τις μεταβλητές αυτές μέσα  
στην συνάρτηση.*/
```

```
global
```

```
score_player_1,score_player_2,card_matches,current_turn,rev  
ealed
```

```
/*Επαναφέρουμε το σκορ του 1 παίκτη στο 0.*/
```

```
score_player_1=0
```

```
/*Επαναφέρουμε το σκορ του δεύτερου παίκτη στο 0.*/
```

```
score_player_2=0
```

```
/*Επαναφέρουμε την σειρά στον πρώτο παίκτη*/
```

```
current_turn=0
```

```
/*Μηδενίζουμε τα ζευγάρια*/
```

```
card_matches=0
```

```
/*κλείνουμε τις κάρτες που είναι ανοιχτές*/
```

```
revealed = [[False] * GRID_SIZE for _ in range(GRID_SIZE)]
```

```
/*Με το print τυπώνουμε ένα μήνυμα στην οθόνη,το οποίο  
παίρνει σαν παράμετρο το μήνυμα που είναι να  
τυπώσουμε.Εδώ πέρα με το print λέμε στον χρήστη ότι το reset  
button πατήθηκε.*/
```

```
print("reset button clicked!")
```

```
/*Ορίζουμε την συνάρτηση new_game(),η οποία είναι για να  
ξεκινήσει νέο παιχνίδι ανακατεύοντας πάλι τις κάρτες.*/
```

```
def new_game():
```

```
/*Ορίζουμε μεταβλητές τύπου global.Αυτό το κάνουμε για να  
μπορούμε να χρησιμοποιήσουμε τις μεταβλητές αυτές μέσα  
στην συνάρτηση.*/
```

```
global
```

```
score_player_1,score_player_2,card_matches,current_turn,rev  
ealed,cards
```

```
/*Επαναφέρουμε το σκορ του 1 παίκτη στο 0.*/
```

```
score_player_1=0
```

```
/*Επαναφέρουμε το σκορ του δεύτερου παίκτη στο 0.*/
```

```
score_player_2=0
```

```

/*Επαναφέρουμε την σειρά στον πρώτο παίκτη*/
current_turn=0

/*Μηδενίζουμε τα ζευγάρια*/
card_matches=0

/*κλείνουμε τις κάρτες που είναι ανοιχτές*/
revealed = [[False] * GRID_SIZE for _ in range(GRID_SIZE)]

/*ανακατεύουμε τις κάρτες και τις βάζουμε στην μεταβλητή
cards.
cards=shuffle_cards()

/* λέμε στον χρήστη ότι το new_game button πατήθηκε.
print("new_game button clicked!")

/*Φτιάχνουμε μια συνάρτηση για να βρούμε την θέση της
κάρτας που επέλεξε κάποιος παίκτης.*/
def get_card_at_pos(pos):

/*Βάζουμε την είσοδο της συνάρτησής get_card_at_position σε
δυο μεταβλητές x και y,για να αποθηκεύσουμε σε αυτές τις
συντεταγμένες του ποντικιού.*/
x,y=pos

```

```
/*Εδώ πέρα φτιάχνουμε ένα for,το οποίο σαρώνει όλες τις γραμμές του κάθε grid.Το row είναι μετρητής και το range(GRID_SIZE) είναι από που ξεκινάει ο μετρητής και που τελειώνει*/
```

```
for row in range(GRID_SIZE):
```

```
/*Εδώ πέρα φτιάχνουμε ένα for,το οποίο σαρώνει όλες τις στήλες του κάθε grid.*/
```

```
for col in range(GRID_SIZE):
```

```
/*Υπολογίζουμε τις συντεταγμένες των οριζόντιων καρτών του κάθε grid.Αυτό γίνεται πολλαπλασιάζοντας τη θέση της κάρτας (στήλη) με το άθροισμα του μεγέθους της κάρτας (CARD_SIZE) και του κενού μεταξύ των καρτών (GAP).Αυτό το κάνουμε για να συγκρίνουμε την κάθε κάρτα στο οριζόντιο άξονα με την συντεταγμένη x του ποντικιού ποιο μετά.*/
```

```
card_x = col * (CARD_SIZE + GAP)
```

```
/*Υπολογίζουμε τις συντεταγμένες των οριζόντιων καρτών του κάθε grid.Αυτό γίνεται πολλαπλασιάζοντας τη θέση της κάρτας (γραμμή) με το άθροισμα του μεγέθους της κάρτας (CARD_SIZE) και του κενού μεταξύ των καρτών (GAP).Αυτό το κάνουμε για να συγκρίνουμε την κάθε κάρτα στο οριζόντιο άξονα με την συντεταγμένη x του ποντικιού ποιο μετά.*/
```

```
card_y = row * (CARD_SIZE + GAP)
```

```
if card_x <= x <= card_x + CARD_SIZE and card_y <= y <= card_y  
+ CARD_SIZE*2
```

```
/*Ελέγχουμε εάν οι συντεταγμένες του ποντικιού x,y βρίσκονται  
μέσα στο πλαίσιο της κάρτας. Συγκεκριμένα ελέγχουμε εάν η  
συντεταγμένη x βρίσκεται μέσα στο μήκος της κάρτας και εάν η  
συντεταγμένη y βρίσκεται μέσα στο πλάτος της κάρτας. Όσον  
αφορά το μήκος της κάρτας θέλουμε η συντεταγμένη x του  
ποντικιού να είναι μεγαλύτερη η ίση από το αριστερό πλάτος  
της κάρτας και μικρότερη η ίση από το αριστερό πλάτος της  
κάρτας συν το μέγεθος της κάρτας(δεξί πλάτος της κάρτας)*/
```

```
/*Η συνάρτηση επιστέφει την γραμμή και την στήλη της  
επιλεγμένης κάρτας*/
```

```
return row, col
```

```
/*Εάν οι άξονες του ποντικιού δεν είναι μέσα στο πλαίσιο της  
κάρτας, τότε η συνάρτηση δεν επιστρέφει τίποτα(None).*/
```

```
return None
```

```
/*Εδώ πέρα παίρνουμε την συνάρτηση Rect της βιβλιοθήκης  
pygame,με την οποία ορίζουμε τις συντεταγμένες ενός  
ορθογωνίου παραλληλογράμμου παίρνει σαν παράμετρο τις  
αρχικές συντεταγμένες x και y ενός ορθογωνίου και τις τελικές  
συντεταγμένες x και y ενός ορθογωνίου. Αυτό το κάνουμε για  
να βάλουμε τις συντεταγμένες του ορθογώνιου  
παραλληλογράμμου σε μια μεταβλητή που λέγεται  
reset_rect*/
```

```
reset_rect=pygame.Rect(900,350,90,40)
```

/*Εδώ πέρα φτιάχνουμε το περίγραμμα του ορθογωνίου του new_game button.

new_game_rect=pygame.Rect(1050,350,160,40)*/

/*Φτιάχνουμε μια Boolean μεταβλητή και τις δίνουμε τιμή true.Αυτήν την μεταβλητή την φτιάχνουμε για να ελέγχουμε το πότε είναι κλειστό και πότε είναι ανοιχτό το παράθυρο pygame. Όταν είναι false,το παράθυρο είναι ανοιχτό.*/

running = True

/*Φτιάχνουμε μια Boolean μεταβλητή, για το πόσο χρόνο θα μένουν ανοιχτές οι κάρτες εάν δεν ταιριάζουν μεταξύ τους.

reveal_time = None

/*φτιάχνουμε μια boolean μεταβλητή η οποία δείχνει αν κάποιος από τους παίκτες έχει επιλέξει την πρώτη κάρτα από το ζευγάρι.*/

first_selection = None

/*φτιάχνουμε μια boolean μεταβλητή η οποία δείχνει αν κάποιος από τους παίκτες έχει επιλέξει την δεύτερη κάρτα από το ζευγάρι.*/

second_selection = None

/*Αρχικοποιούμε το σκορ του 1 παίκτη στο 0.*/

score_player_1=0

```
/*Αρχικοποιούμε το σκορ του δεύτερου παίκτη στο 0.*/  
score_player_2=0
```

```
/*Αρχικοποιούμε την σειρά στον πρώτο παίκτη*/  
current_turn=0
```

```
/*Αρχικοποιούμε τα ζευγάρια στο 0*/  
card_matches=0
```

```
/*Ορίζουμε έναν βρόγχο while,το οποίο θα εκτελείται όσο αυτό  
που είναι δεξιά από το while είναι true.Εδώ λέμε ότι όσο το  
παράθυρο pygame είναι ανοιχτό, θα εκτελείται */
```

```
while running:
```

```
/*Εδώ πέρα παίρνουμε την μεταβλητή (screen =  
pygame.display.set_mode((SCREEN_SIZE*1.7, SCREEN_SIZE))  
και πάνω σε αυτή εφαρμόζουμε την συνάρτηση fill,η οποία έχει  
σαν παράμετρο το χρώμα που είναι να εφαρμόσουμε σε μια  
επιφάνεια. Στην συγκεκριμένη περίπτωση γεμίζουμε την  
επιφάνεια του παραθύρου pygame με άσπρο χρώμα */
```

```
screen.fill(BACKGROUND_COLOR)
```

```
/*Εδώ πέρα έχουμε την get του module event,η οποία  
επιστρέφει μια λίστα με όλα τα event που έχουν συμβεί μέχρι  
αυτή την χρονική στιγμή και δεν παίρνει καμία  
είσοδο(παράμετρο).Μετά διατρέχουμε τα event από την λίστα
```

ένα-ένα και τα αποθηκεύουμε προσωρινά στην μεταβλητή ev για επεξεργασία.*/

```
for ev in pygame.event.get():
```

```
/* Εδώ πέρα το type είναι ένα attribute του module event.Ελέγχουμε εάν κάποια από τα event που έχουν συμβεί είναι το κλείσιμο του παραθύρου pygame(QUIT).*/
```

```
if ev.type == pygame.QUIT:
```

```
/*Κλείνουμε το παράθυρο pygame,αλλάζοντας την τιμή του running στο False.*/
```

```
running = False
```

```
/*Εδώ πέρα το MOUSEBUTTONDOWN είναι κάποιο event που συμβαίνει όταν πατηθεί κάποιο κουμπί του ποντικιού.Όποτε ελέγχουμε εάν πατήθηκε κάποιο κουμπί στο ποντίκι.*/
```

```
elif ev.type == pygame.MOUSEBUTTONDOWN:
```

```
/*Εδώ πέρα έχουμε την get_pos() του module της βιβλιοθήκης pygame,η οποία βρίσκει τις συντεταγμένες του ποντικιού αυτή την χρονική στιγμή και δεν παίρνει κάποια είσοδο για παράμετρο.*/
```

```
pos = pygame.mouse.get_pos()
```

```
/*Εδώ πέρα βάζουμε σαν είσοδο στην συνάρτηση get_card_at_pos τις συντεταγμένες του ποντικιού και μετά
```

βάζουμε την συνάρτηση σε μια μεταβλητή με το όνομα
card_pos.*/

```
card_pos = get_card_at_pos(pos)
```

/*Εδώ έχουμε την συνάρτηση collidepoint της βιβλιοθήκης
pygame,την οποία εφαρμόζουμε πάνω στο πλαίσιο του reset,η
οποία ελέγχει εάν κάποιο point βρίσκεται μέσα σε ένα
rectahgle. Αυτή η συνάρτηση παίρνει δυο παραμέτρους, τις
συντεταγμένες x και y του point.Αυτό το κάνουμε για να
ελέγξουμε εάν οι συντεταγμένες του ποντικιού βρίσκονται μέσα
στο πλαίσιο του reset button.*/

```
if reset_rect.collidepoint(pos):
```

/*Εκτελούμε την συνάρτηση reset.*/

```
reset()
```

/*Ελέγχουμε εάν οι συντεταγμένες του ποντικιού βρίσκονται
μέσα στο πλαίσιο του reset button.*/

```
elif new_game_rect.collidepoint(pos):
```

/*Εκτελούμε την συνάρτηση new game.*/

```
new_game()
```

/*Ελέγχουμε εάν η μεταβλητή card_pos,στην έχουμε
αποθηκεύσει την συνάρτηση get_card_at_pos έχει κάποια τιμή
και εάν η κάρτα στην συγκεκριμένη
θέση([card_pos[0][card_pos[1]]) δεν είναι ανοιχτή.*/

```
elif card_pos and not revealed[card_pos[0]][card_pos[1]]:
```

```
/*ελέγχουμε εάν ο παίκτης δεν έχει επιλέξει την πρώτη κάρτα  
από το ζευγάρι.*/
```

```
if first_selection is None:
```

```
/*Αποθηκεύουμε την τιμή της μεταβλητής card_pos στην  
μεταβλητή first_selection,η οποία ήταν false.*/
```

```
first_selection = card_pos
```

```
/*Ανοίγουμε την κάρτα στην θέση card_pos[0],card_pos[1].*/
```

```
revealed[card_pos[0]][card_pos[1]] = True
```

```
/*ελέγχουμε εάν ο παίκτης δεν έχει επιλέξει την δεύτερη κάρτα  
από το ζευγάρι.*/
```

```
elif second_selection is None:
```

```
/*Αποθηκεύουμε την τιμή της μεταβλητής card_pos στην  
μεταβλητή second_selection,η οποία ήταν false.*/
```

```
second_selection = card_pos
```

```
/*Ανοίγουμε την κάρτα στην καινούργια θέση  
card_pos[0],card_pos[1].*/
```

```
revealed[card_pos[0]][card_pos[1]] = True
```

/*Εδώ πέρα έχουμε την συνάρτηση get_ticks του module time της βιβλιοθήκης pygame, η οποία δεν παίρνει καμιά παράμετρο σαν είσοδο. Αυτή η συνάρτηση σταματάει το τρέξιμο του προγράμματος και αρχίζει να μετράει χρόνο για κάποια δευτερόλεπτα χρησιμοποιώντας τον processor. Αυτό το χρόνο μετά το αποθηκεύει σε μια μεταβλητή με όνομα reveal_time. Αυτό το κάνουμε για να αρχίσουμε να μετράμε χρόνο στον οποίο θα παραμένουν ανοιχτές οι κάρτες πριν κλείσουν.*/*

```
reveal_time = pygame.time.get_ticks()
```

/*Εδώ πέρα ελέγχουμε εάν η μεταβλητή reveal-time έχει κάποια τιμή(if reveal_time) και ελέγχουμε εάν ο τωρινός χρόνος πλην τον χρόνο που αποκαλύφθηκε το ζευγάρι είναι μεγαλύτερο από 150 δευτερόλεπτα. Δηλαδή στην περίπτωση που δεν υπάρχει ζευγάρι, μετράμε εάν έχουν περάσει 150 δευτερόλεπτα από τότε που άνοιξαν οι κάρτες..*/*

```
if reveal_time and pygame.time.get_ticks() - reveal_time > 150:
```

/*Ελέγχουμε εάν έχει επιλεγεί κάποιο ζευγάρι. Το and εδώ πέρα είναι ένωση δυο συνθηκών(if first_selection, if second_selection) και πρέπει να ισχύουν και οι δυο συνθήκες ταυτοχρόνως για να εκτελεστεί αυτό που είναι κάτω από το if.*/*

```
if first_selection and second_selection:
```

/*Βάζουμε τις τιμές της μεταβλητής first_selection, στην οποία αποθηκεύσαμε τις συντεταγμένες x και y της επιλεγμένης κάρτας σε δυο μεταβλητές r1,c1. Έτσι στην r1 θα βάλουμε την

συντεταγμένη x και στην c1 την συντεταγμένη y της επιλεγμένης κάρτας.*/

r1,c1=first_selection

/*Βάζουμε τις τιμές της μεταβλητής first_selection,στην οποία αποθηκεύσαμε τις συντεταγμένες x και y της επιλεγμένης κάρτας σε δυο μεταβλητές r2,c2.Έτσι στην r2 θα βάλουμε την συντεταγμένη x και στην c2 την συντεταγμένη y της επιλεγμένης κάρτας.*/

r2,c2=first_selection

/*Ελέγχουμε εάν υπάρχει ζευγάρι στις θέσεις r1,c1 και r2,c2.Το == είναι τελεστής σύγκρισης ισότητας.*/

if cards[r1][c1]==cards[r2][c2]:

/* Ελέγχουμε εάν η διαίρεση της τιμής της μεταβλητής current_turn με το 2 είναι ίσο με το μηδέν.Αυτό το κάνουμε για να ελέγξουμε εάν είναι η σειρά του παίκτη 1 να παίξει.Η μεταβλητή current_turn ξεκινάει από το μηδέν.*/

if current_turn%2==0:

/*Αυξάνουμε το σκορ του παίκτη 1 κατά 1*/

score_player_1=score_player_1+1

/*Αφήνουμε την κάρτα του ζευγαριού στην θέση r1,c1 ανοιχτή για όλη την διάρκεια του παιχνιδιού*/

revealed[r1][c1]=True

```
/*Αφήνουμε την κάρτα του ζευγαριού στην θέση r2,c2  
ανοιχτή για όλη την διάρκεια του παιχνιδιού*/
```

```
revealed[r2][c2]=True
```

```
/*Αυξάνουμε τα συνολικά ζευγάρια κατά 1*/
```

```
card_matches=card_matches+1
```

```
/*Ελέγχουμε εάν είναι η σειρά του δεύτερου παίκτη να παίξει.  
Η μεταβλητή current_turn έχει αρχική τιμή 0*/
```

```
elif current_turn%2==1
```

```
/*Αυξάνουμε το σκορ του παίκτη 2 κατά 1*/
```

```
score_player_2=score_player_2+1
```

```
/*Αφήνουμε την κάρτα του ζευγαριού στην θέση r1,c1  
ανοιχτή για όλη την διάρκεια του παιχνιδιού*/
```

```
revealed[r1][c1]=True
```

```
/*Αφήνουμε την κάρτα του ζευγαριού στην θέση r2,c2  
ανοιχτή για όλη την διάρκεια του παιχνιδιού*/
```

```
revealed[r2][c2]=True
```

```
/*Αυξάνουμε τα συνολικά ζευγάρια κατά 1*/
```

```
card_matches=card_matches+1
```

```
/*Είναι η περίπτωση που δεν βρέθηκε ζευγάρι.*/
```

```
else:
```

```
/*Αύξηση της σειράς κατά 1*/
```

```
current+turn=current_turn+1
```

```
/*κλείνουμε την κάρτα στην θέση r1,c1
```

```
revealed[r1][c1]=False
```

```
/*κλείνουμε την κάρτα στην θέση r2,c2
```

```
revealed[r2][c2]=False
```

```
/*Αντικαθιστούμε την τιμή της μεταβλητής first_selection με  
None για να μπορεί ο παίκτης να επιλέξει νέα πρώτη κάρτα για  
το επόμενο ζευγάρι.*/
```

```
first_selection=None
```

```
/*Αντικαθιστούμε την τιμή της μεταβλητής second_selection με  
None για να μπορεί ο παίκτης να επιλέξει νέα δεύτερη κάρτα  
για το επόμενο ζευγάρι.*/
```

```
first_selection=None
```

```
/*Εδώ πέρα φτιάχνουμε ένα for, το οποίο σαρώνει όλες τις  
γραμμές του κάθε grid. Το row είναι μετρητής και το
```

range(GRID_SIZE) είναι από που ξεκινάει ο μετρητής και που τελειώνει*/

```
for row in range(GRID_SIZE):
```

/*Εδώ πέρα φτιάχνουμε ένα for, το οποίο σαρώνει όλες τις στήλες του κάθε grid.*/

```
for col in range(GRID_SIZE):
```

/*Υπολογίζουμε τις συντεταγμένες x της κάθε κάρτας του κάθε grid για να την σχεδιάσουμε. Αυτό γίνεται πολλαπλασιάζοντας τη θέση της κάρτας (στήλη) με το άθροισμα του μεγέθους της κάρτας (CARD_SIZE) και του κενού μεταξύ των καρτών (GAP). Μετά τις συντεταγμένες αυτές τις βάζουμε σε μια μεταβλητή που λέγεται */

```
x = col * (CARD_SIZE + GAP)
```

/*Υπολογίζουμε τις συντεταγμένες y της κάθε κάρτας του κάθε grid για να την σχεδιάσουμε. Αυτό γίνεται πολλαπλασιάζοντας τη θέση της κάρτας (γραμμή) με το άθροισμα του μεγέθους της κάρτας (CARD_SIZE) και του κενού μεταξύ των καρτών (GAP). Μετά τις συντεταγμένες αυτές τις βάζουμε σε μια μεταβλητή που λέγεται y.*/

```
y = row * (CARD_SIZE + GAP)
```

/*Ελέγχουμε εάν η κάρτα στο συγκεκριμένο row,col είναι ανοιχτή*/

```
if revealed[row][col]:
```

/*Εδώ πέρα έχουμε την συνάρτηση rect του module draw της βιβλιοθήκης pygame, η οποία ζωγραφίζει ένα ορθογώνιο παραλληλόγραμμο και παίρνει σαν πρώτο όρισμα την επιφάνεια που θα ζωγραφιστεί το ορθογώνιο παραλληλόγραμμο, σαν δεύτερο όρισμα το χρώμα του ορθογωνίου παραλληλογράμμου, τις αρχικές και τελικές συντεταγμένες του ορθογωνίου παραλληλογράμμου και τέλος το πάχος της γραμμής του ορθογωνίου παραλληλογράμμου.

Άρα εδώ φτιάχνουμε ένα ορθογώνιο παραλληλόγραμμο με μαύρο περίγραμμα(0,0,0) στην θέση x και y, το οποίο θα έχει μήκος 100 και πλάτος 200 και πάχος περιγράμματος 5.* /

```
pygame.draw.rect(screen, (0, 0, 0), (x, y, CARD_SIZE, CARD_SIZE * 2), 5)
```

/*Εδώ πέρα έχουμε την συνάρτηση Font της βιβλιοθήκης pygame, η οποία παίρνει ένα font από κάποιο φάκελο και δημιουργεί ένα καινούργιο font και παίρνει σαν πρώτο όρισμα το file path και σαν δεύτερο όρισμα την γραμματοσειρά. Άρα εδώ παίρνουμε το font που υπάρχει στο έτοιμο αρχείο freesansbold.ttf(freesan γραμματοσειρά με bold γράμματα) που υπάρχει στην python και του βάζουμε μέγεθος γραμματοσειράς 32. Αυτήν την συνάρτηση μετά την βάζουμε σε μια μεταβλητή που λέγεται font.* /

```
font = pygame.font.Font('freesansbold.ttf', 32)
```

/*Εδώ πέρα εφαρμόζουμε την συνάρτηση render της βιβλιοθήκης pygame πάνω στο font που φτιάξαμε, η οποία ορίζει κάποιο κείμενο και παίρνει σαν πρώτο όρισμα το κείμενο που θέλουμε να εμφανίσουμε στην οθόνη, σαν δεύτερο όρισμα

μια Boolean τιμή που λειει εάν θέλουμε το κείμενο να είναι ποιο λείο και καθαρό, σαν τρίτο όρισμα το χρώμα του κειμένου σε μορφή `rbg(red,green,blue)` και σαν τέταρτο όρισμα το `background color` του κειμένου. Αυτήν την συνάρτηση μετά την βάζουμε σε μια μεταβλητή που λέγεται `text`.

Άρα εδώ ορίζουμε ένα κείμενο, το οποίο θα είναι τα ανακατεμένα σύμβολα που επιστέφει η συνάρτηση `card_shuffle` με κόκκινο χρώμα(255,0,0).*/

```
text = font.render(cards[row][col], False, (255, 0, 0))
```

/*φτιάχνουμε μια λίστα που περιέχει όλες τις θέσεις που θα μπουν το κάθε σύμβολο(A,B,C,D,E,H,K,F) που είναι μέσα στην λίστα με τα ανακατεμένα σύμβολα που επιστέφει η συνάρτηση `card_shuffle`. Κάθε σύμβολο τοποθετείται ξεχωριστά σε κάθε κάρτα.*/

```
positions = [
```

```
(x + 15, y + 15), #Πάνω αριστερά
```

```
(x + CARD_SIZE - 15, y + 15), #Πάνω δεξιά
```

```
(x + CARD_SIZE // 2 - 15, y + CARD_SIZE // 2+15),#κέντρο πάνω αριστερά
```

```
(x + CARD_SIZE // 2+15, y + CARD_SIZE // 2+15 ),#κέντρο πάνω δεξιά
```

```
(x + CARD_SIZE // 2 - 15, y + CARD_SIZE // 2+45),#κέντρο κέντρο αριστερά
```

```
(x + CARD_SIZE // 2 +15, y + CARD_SIZE // 2+45),#κέντρο κέντρο δεξιά
```

(x + CARD_SIZE // 2 - 15, y + CARD_SIZE // 2+75),#κέντρο κάτω
αριστερά

(x + CARD_SIZE // 2 + 15, y + CARD_SIZE // 2+75),#κέντρο κάτω
δεξιά

(x + 15, y + 175),#κάτω αριστερά

(x + CARD_SIZE-15, y + 175),]#κάτω δεξιά

/*Βάζουμε μια λούπα for,η οποία θα διατρέξει από μια φόρα
την λίστα positions*/

for pos in positions:

/*Εδώ πέρα έχουμε την συνάρτηση get_rect της βιβλιοθήκης
pygame η οποία ζωγραφίζει ένα ορθογώνιο παραλληλόγραμμο
γύρω από μια επιφάνεια και παίρνει σαν όρισμα την θέση του
κέντρου του ορθογωνίου παραλληλογράμμου. Άρα
δημιουργούμε ένα πλαίσιο γύρω από τα σύμβολα των
καρτών.*/

text_rect = text.get_rect(center=pos)

/*Εδώ εφαρμόζουμε την συνάρτηση blit πάνω στην τιμή της
μεταβλητής που έχουμε ορίσει, η οποία ζωγραφίζει μια source
εικόνα πάνω σε μια άλλη εικόνα και παίρνει σαν πρώτο όρισμα
την source εικόνα και σαν δεύτερο όρισμα την destination
εικόνα. Εδώ πέρα source εικόνα είναι η μεταβλητή text και
destination εικόνα text_rect.*/

screen.blit(text, text_rect)

```
/*όταν οι κάρτες είναι κλειστές*/
```

```
else:
```

```
/*Δημιουργούμε ένα κόκκινο ορθογώνιο παραλληλόγραμμο  
με αρχικές συντεταγμένες x,y( $x = col * (CARD\_SIZE + GAP)$ ), $y =$   
 $row * (CARD\_SIZE + GAP)$ ),μήκος  $CARD\_SIZE=100$ ,πλάτος  
 $CARD\_SIZE*2=100*2=200$  και πάχος γραμμής 0.*/
```

```
pygame.draw.rect(screen, (255, 0, 0), (x, y, CARD_SIZE,  
CARD_SIZE * 2), 0)
```

```
/*Δημιουργούμε ένα κόκκινο ορθογώνιο παραλληλόγραμμο με  
μήκος 500 και πλάτος 100, με αρχικές συντεταγμένες x,y 780 και  
50 και πλάτος γραμμής 0 για το σκορ του κάθε παίκτη.*/
```

```
pygame.draw.rect(screen,(150,0,0),(780,50,500,100),0)
```

```
/*ορίζουμε ένα καινούργιο font*/
```

```
font = pygame.font.Font('freesansbold.ttf', 32)
```

```
/*Εδώ ορίζουμε ένα κείμενο, το οποίο θα λει score player 1.Να  
θυμίσουμε ότι το False σημαίνει ότι το κείμενό μας δεν θέλουμε  
να είναι ποιο λείο και καθαρό.*/
```

```
texts1 = font.render('score player 1', False, (255, 0, 0))
```

```
/*Δημιουργούμε ένα πλαίσιο γύρω από το κείμενο που λείι  
score player 1,το οποίο θα έχει κέντρο το σημείο 900,70.*/
```

```
text_rects1 = texts1.get_rect(center=(900,70))
```

```
/*Βάζουμε το κείμενο score player 1 μέσα στο text_rects1*/  
screen.blit(texts1, text_rects1)
```

```
/*Δημιουργούμε μια γραμμή με την συνάρτηση line του  
module draw της βιβλιοθήκης pygame,η οποία παίρνει σαν  
πρώτο όρισμα την επιφάνεια πάνω στην οποία θα σχεδιάσει την  
γραμμή(screen),το χρώμα της γραμμής(0,0,0),τις αρχικές  
συντεταγμένες(1025,50),τις τελικές συντεταγμένες(1025,150)  
και το πάχος της γραμμής(5).Αυτό το κάνουμε για το  
διαχωρισμό των σκορ των δυο παικτών.*/
```

```
pygame.draw.line(screen, (0,0,0), (1025, 50), (1025, 150), 5)
```

```
/*Εδώ ορίζουμε ένα κείμενο,το οποίο θα λει score player 2.Να  
θυμίσουμε ότι το False σημαίνει ότι το κείμενό μας δεν θέλουμε  
να είναι ποιο λείο και καθαρό.*/
```

```
Texts2 = font.render('score player 2', False, (255, 0, 0))
```

```
/*Δημιουργούμε ένα πλαίσιο γύρω από το κείμενο που λείει  
score player 2,το οποίο θα έχει κέντρο το σημείο 1155,70.*/
```

```
text_rects2 = texts2.get_rect(center=(1155,70))
```

```
/*Βάζουμε το κείμενο score player 2 μέσα στο text_rects2*/  
screen.blit(texts2, text_rects2)
```

```
/*Δημιουργούμε ένα άσπρο ορθογώνιο μέσα στο οποίο θα  
είναι το σκορ του πρώτου παίκτη.*/
```

```
pygame.draw.rect(screen,(255,255,255),(800,80,205,60),0)
```

```
/*Δημιουργούμε ένα άλλο άσπρο ορθογώνιο μέσα στο οποίο  
θα είναι το σκορ του δεύτερου παίκτη.*/
```

```
pygame.draw.rect(screen,(255,255,255),(1050,80,205,60),0)
```

```
/*Εδώ ορίζουμε ένα κείμενο, το οποίο θα λέει την τιμή του score  
του player 1.Το string που έχουμε βάλει εδώ μετατρέπει την  
τιμή της μεταβλητής scor_player_1 σε string,γιατί το πρώτο  
όρισμα της συνάρτησης είναι πάντα τύπου string. Να  
θυμίσουμε ότι το False σημαίνει ότι το κείμενό μας δεν θέλουμε  
να είναι ποιο λείο και καθαρό.*/
```

```
textns1 = font.render(str(score_player_1), False, (255, 0, 0))
```

```
/*Δημιουργούμε ένα πλαίσιο γύρω από την τιμή της  
μεταβλητής textns1, το οποίο θα έχει κέντρο το σημείο  
900,110.*/
```

```
text_rectns1 = textns1.get_rect(center=(900,110))
```

```
/*Βάζουμε την τιμή της μεταβλητής textns1 μέσα στο  
text_rectns1*/
```

```
screen.blit(textns1, text_rectns1)
```

```
/*Εδώ ορίζουμε ένα κείμενο, το οποίο θα λέει την τιμή του score  
του player 2.Το string που έχουμε βάλει εδώ μετατρέπει την
```

τιμή της μεταβλητής scor_player_2 σε string,γιατί το πρώτο όρισμα της συνάρτησης είναι πάντα τύπου string. Να θυμίσουμε ότι το False σημαίνει ότι το κείμενό μας δεν θέλουμε να είναι ποιο λείο και καθαρό.*/

```
textns2 = font.render(str(score_player_2), False, (255, 0, 0))
```

/*Δημιουργούμε ένα πλαίσιο γύρω από την τιμή της μεταβλητής textns2,το οποίο θα έχει κέντρο το σημείο 1155,110.*/

```
text_rectns2 = textns2.get_rect(center=(1155,110))
```

/*Βάζουμε την τιμή της μεταβλητής textns2 μέσα στο text_rectns2*/

```
screen.blit(textns2, text_rectns2)
```

/*Δημιουργούμε ένα κείμενο ,το οποίο θα λέει ποιανού σειρά είναι να παίξει. Το f σημαίνει f-string,δηλαδή ένας ειδικός τύπος string το οποίο μπορεί να περιέχει και κάτι άλλο εκτός από χαρακτήρες μόνο μέσα στα "" */.

```
turn_text = font.render(f"{current_turn % 2 + 1}'s Turn", False, (0, 153, 0))
```

/*Δημιουργούμε ένα πλαίσιο γύρω από την τιμή της μεταβλητής turn_text,το οποίο θα έχει κέντρο το σημείο 1030,250.*/

```
turn_text_rect = turn_text.get_rect(center=(1030, 250))
```

```
/*Βάζουμε την τιμή της μεταβλητής turn_text μέσα στο  
πλαίσιο turn_text_rect*/
```

```
screen.blit(turn_text, turn_text_rect)
```

```
/*Ζωγραφίζουμε το reset button στην οθόνη. Στην θέση του  
τρίτου ορίσματος της συνάρτησης rect βάζουμε την μεταβλητή  
reset_rect,οπού έχουμε αποθηκεύσει τις συντεταγμένες του  
ορθογωνίου παραλληλογράμμου του reset button*/
```

```
pygame.draw.rect(screen,(96,96,96),reset_rect)
```

```
/*Εδώ ορίζουμε ένα κείμενο, το οποίο θα λέει reset.Να  
θυμίσουμε ότι το False σημαίνει ότι το κείμενό μας δεν θέλουμε  
να είναι ποιο λείο και καθαρό.*/
```

```
textbure = font.render('reset', False, (255, 0, 0))
```

```
/*Δημιουργούμε ένα πλαίσιο γύρω από το κείμενο που λέει  
reset,το οποίο θα έχει κέντρο το σημείο 940,370.*/
```

```
textbure_rect = textbure.get_rect(center=(940,370))
```

```
/*Βάζουμε το κείμενο reset μέσα στο πλαίσιο textbure_rect*/
```

```
screen.blit(textbure, textbure_rect)
```

```
/*Ζωγραφίζουμε το new game button στην οθόνη. Στην θέση  
του τρίτου ορίσματος της συνάρτησης rect βάζουμε την  
μεταβλητή new_game_rect, οπού έχουμε αποθηκεύσει τις  
συντεταγμένες του ορθογωνίου παραλληλογράμμου του new  
game button*/
```

```
pygame.draw.rect(screen,(96,96,96),new_game_rect)
```

```
/*Εδώ ορίζουμε ένα κείμενο, το οποίο θα λέει new game.Να  
θυμίσουμε ότι το False σημαίνει ότι το κείμενό μας δεν θέλουμε  
να είναι ποιο λείο και καθαρό.*/
```

```
textbung = font.render('new game', False, (255, 0, 0))
```

```
/*Δημιουργούμε ένα πλαίσιο γύρω από το κείμενο που λέει  
new game,το οποίο θα έχει κέντρο το σημείο 1130,370.*/
```

```
textbung_rect = textbung.get_rect(center=(1130,370))
```

```
/*Βάζουμε το κείμενο new game μέσα στο πλαίσιο  
textbure_rect*/
```

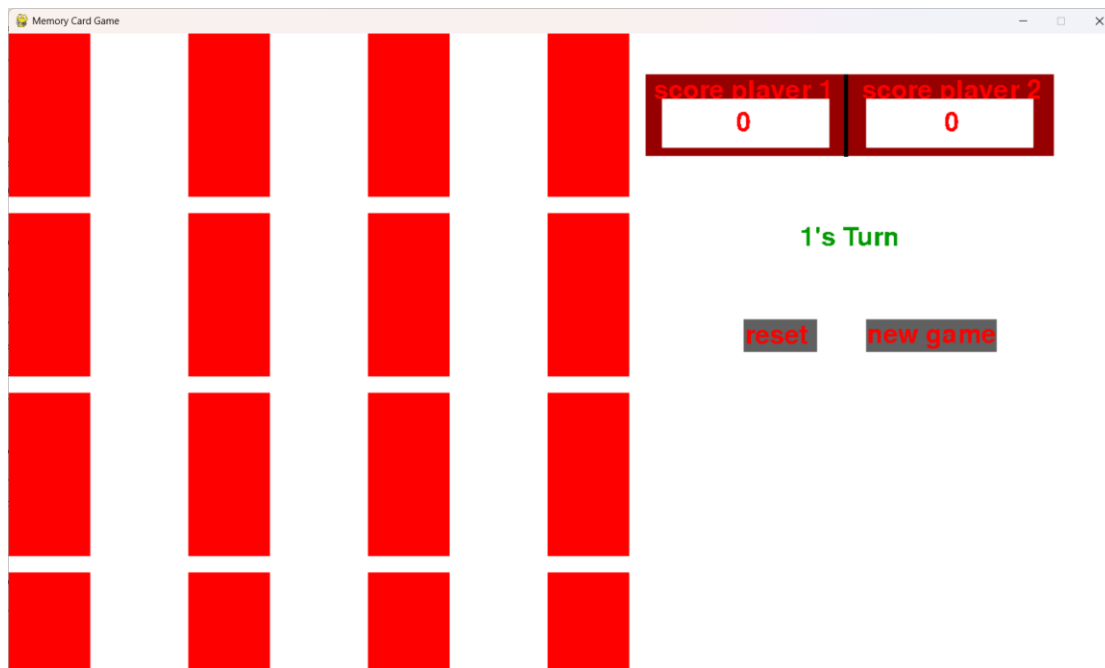
```
screen.blit(textbung, textbung_rect)
```

```
/*Εδώ πέρα έχουμε την συνάρτηση update του module  
display της βιβλιοθήκης pygame, η οποία δεν παίρνει κάποια  
παράμετρο για είσοδο και ενημερώνει το παράθυρο  
pygame.*/
```

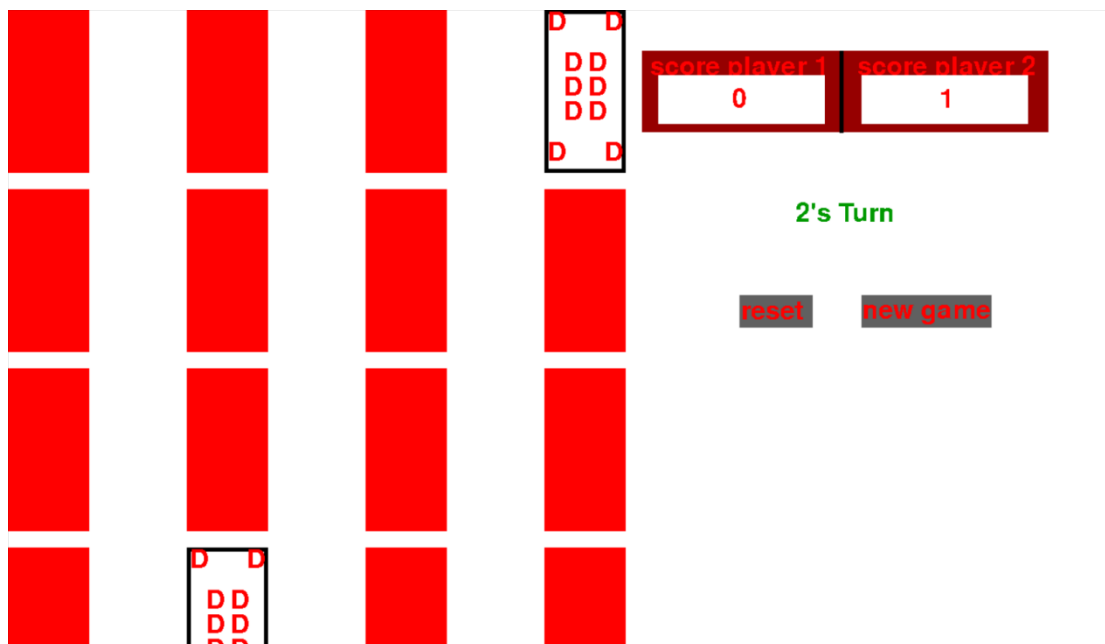
```
pygame.display.update()
```

.

3.Εικόνες από το παιχνίδι:



Εικόνα 8:περιβάλλον παιχνιδιού



Εικόνα 9:περιβάλλον παιχνιδιού σε λειτουργία

Συμπεράσματα

Σε αυτήν την διπλωματική εργασία, δημιούργησα ένα παιχνίδι με κάρτες μνήμης χρησιμοποιώντας την Python και τη βιβλιοθήκη Pygame. Στην αρχή, μίλησα λίγο για την Python — τι είναι, γιατί είναι εύκολη στην εκμάθηση και πού μπορεί να χρησιμοποιηθεί, όπως στην επιστήμη δεδομένων ή την τεχνητή νοημοσύνη. Στη συνέχεια, εξήγησα τι είναι ένας διερμηνέας και πώς βοηθά στην εκτέλεση προγραμμάτων Python. Το κύριο μέρος ήταν το ίδιο το παιχνίδι: οι παίκτες επιλέγουν δύο κάρτες, προσπαθούν να βρουν ζεύγη και συλλέγουν πόντους. Οι κάρτες ανακατεύονται στην αρχή κάθε γύρου και το παιχνίδι συνεχίζεται μέχρι να βρεθούν όλα τα ζεύγη.

Τελικά, αυτό το έργο δείχνει ότι η Python είναι μια απλή και ευέλικτη γλώσσα που μπορεί να χρησιμοποιηθεί όχι μόνο για μεγάλα, επαγγελματικά έργα αλλά και για διασκεδαστικά και εκπαιδευτικά παιχνίδια. Δημιουργώντας αυτό το παιχνίδι, έμαθα πώς να γράφω πιο καθαρό κώδικα, να λύνω προβλήματα βήμα προς βήμα και να χρησιμοποιώ τον προγραμματισμό με δημιουργικό τρόπο. Η εργασία αποδεικνύει ότι η Python είναι ένα εξαιρετικό εργαλείο για αρχάριους και προχωρημένους χρήστες για να ζωντανέψουν ιδέες.

Πηγες:

- [1] https://www.tutorialspoint.com/python/python_history.htm
- [2] https://arch.ece.uowm.gr/docs/Python_Programming_Full_Book_Dasygenis_Terzidou.pdf
- [3] <https://www.geeksforgeeks.org/python/python-language-advantages-applications/>
- [4] <https://www.tokioschool.com/en/news/main-characteristics-python/>
- [5] <https://www.simplilearn.com/future-of-python-article>
- [6] <https://www.tutorchase.com/answers/a-level/computer-science/what-are-the-stages-of-interpreting-a-programming-language>
- [7] [https://el.wikipedia.org/wiki/%CE%94%CE%B9%CE%B5%CF%81%CE%BC%CE%B7%CE%BD%CE%AD%CE%B1%CF%82_\(%CF%85%CF%80%CE%BF%CE%BB%CE%BF%CE%B3%CE%B9%CF%83%CF%84%CE%AD%CF%82\)](https://el.wikipedia.org/wiki/%CE%94%CE%B9%CE%B5%CF%81%CE%BC%CE%B7%CE%BD%CE%AD%CE%B1%CF%82_(%CF%85%CF%80%CE%BF%CE%BB%CE%BF%CE%B3%CE%B9%CF%83%CF%84%CE%AD%CF%82))
- [8] https://www.geeksforgeeks.org/compiler-design/introduction-of-lexical-analysis/?utm_source=chatgpt.com
- [9] <https://www.emertxe.com/embedded-systems/c-programming/c-projects/lexical-analyzer/>
- [10] https://www.uobabylon.edu.iq/eprints/publication_12_17490_344.pdf
- [11] https://en.wikipedia.org/wiki/Abstract_syntax_tree
- [12] https://www.geeksforgeeks.org/theory-of-computation/what-is-context-free-grammar/?utm_source=chatgpt.com
- [13] <https://www.geeksforgeeks.org/nlp/understanding-semantic-analysis-nlp/>
- [14] <https://www.shiksha.com/online-courses/articles/interpreter-types-advantages-and-disadvantages/>
- [15] https://medium.com/@annie_/various-interpreters-27ba8a236e00
- [16] <https://www.digitalfutureaotearoa.nz/digital-pikau/python>
- [17] <https://www.wscubetech.com/resources/python/advantages>
- [18] <https://www.educba.com/python-features/>
- [19] https://www.youtube.com/watch?v=txr8HNF_i_6I
- [20] <https://teachcomputerscience.com/syntax-analysis/>