



Πανεπιστήμιο Πειραιώς

ΠΜΣ ΠΛΗΡΟΦΟΡΙΚΗ

4ος Κύκλος/ Β' ΕΞΑΜΗΝΟ

Τεχνητή Νοημοσύνη και Έμπειρα Συστήματα

Αλγόριθμος NEAT σε παιχνίδι 2D

ΚΩΝΣΤΑΝΤΙΝΟΣ ΚΟΛΙΟΣ

ΜΠΠΑ 21032

20.08.2022

Περιεχόμενα

Εισαγωγή	2
Τι Είναι τα Νευρωνικά Δίκτυα;	3
Πως Εκπαιδεύουμε ένα Νευρωνικό Δίκτυο	4
Πλεονεκτήματα των Νευρωνικών Δικτύων	7
Παραδείγματα νευρωνικών δικτύων στην καθημερινή ζωή	8
Σκοπός της Εργασίας	9
Εγκατάσταση της Python	10
Εγκατάσταση του PyCharm	13
Δημιουργία νέου Project μέσω του PyCharm και Κλωνοποίηση μέσω GitHub	17
Εγκατάσταση των βιβλιοθηκών pygame και neat	21
Configurations που αφορούν τον NEAT:	22
Λεπτομέρειες για το Παιχνίδι και Βίντεο επεξήγησης της υλοποίησης:	24

Εισαγωγή

Τι Είναι τα Νευρωνικά Δίκτυα;

Τα νευρωνικά δίκτυα είναι μια σχετικά πρόσφατη περιοχή στις θετικές επιστήμες διότι έχουν αναπτυχθεί και μελετηθεί τα τελευταία 40 χρόνια. Βασίζονται στο νευρικό σύστημα των ζώντων οργανισμών, ωστόσο η μελέτη δεν περιλαμβάνει μόνο βιολογικούς οργανισμούς αλλά τη λύση κάθε είδους προβλημάτων με την χρήση ηλεκτρονικού υπολογιστή συνδυάζοντας τον τρόπο σκέψης του ανθρώπινου εγκεφάλου με τον αφηρημένο μαθηματικό τρόπο σκέψης (π.χ. ένα δίκτυο μπορεί να μάθει και να εκπαιδευτεί ώστε να επιλύσει ένα πρόβλημα που πολλές φορές θα είναι και η βέλτιστη λύση, διότι επεξεργάζεται διάφορους παράγοντες που το ανθρώπινο μυαλό αδυνατεί να τους εντοπίσει).

Ένα ιδιαίτερο χαρακτηριστικό είναι ότι οι επιστήμονες στην περιοχή των νευρωνικών δικτύων προέρχονται σχεδόν από όλους τους κλάδους των φυσικών επιστημών, όπως την ιατρική, την επιστήμη μηχανικών, τη φυσική, τη χημεία, τα μαθηματικά, την επιστήμη υπολογιστών κτλ., που καμία άλλη επιστήμη σήμερα δεν συνδυάζει γνώσεις από τόσο διαφορετικές περιοχές με τόσο άμεσο τρόπο.

Η έμπνευση για κάθε μορφή νευρωνικού δικτύου ξεκινά από την βιολογία. Οι ζώντες οργανισμοί, από τους πιο απλούς μέχρι τον άνθρωπο, έχουν ένα νευρικό σύστημα, το οποίο είναι υπεύθυνο για πολλές διεργασίες όπως τη μάθηση, τη μνήμη, την κίνηση κτλ. και η διεκπεραίωση των διεργασιών οφείλεται στον εγκέφαλο, που αποτελεί ένα νευρωνικό δίκτυο. Συνεπώς ένα νευρωνικό δίκτυο αποτελείται από νευρώνες που στην ουσία στέλνουν ηλεκτρικά σήματα σε άλλους νευρώνες, όπως ακριβώς λειτουργεί ο ανθρώπινος εγκέφαλος. Για αυτόν

τον λόγο για την καλύτερη κατανόηση τους, απαιτείται η εμβάθυνση στην επιστήμη της βιολογίας και της ιατρικής διότι τα πρώτα νευρωνικά δίκτυα που μελετήθηκαν οφείλονται στις παραπάνω επιστήμες και ονομάζονται βιολογικά νευρωνικά δίκτυα.

Εκτός από τα βιολογικά νευρωνικά δίκτυα υπάρχουν και τα τεχνητά νευρωνικά δίκτυα που η βασική τους διαφορά είναι ότι τα δίκτυα αυτά μαθαίνουν με την εξάσκηση και την εμπειρία, όπως οι άνθρωποι, άλλα διαφέρουν στο ότι δεν ακολουθούν προκαθορισμένους κανόνες που είναι χαρακτηριστικό των υπολογιστών. Ο σκοπός της λειτουργίας ενός τεχνητού νευρωνικού δικτύου είναι να εκτελεί διεργασίες μόνο του πχ. να μπορεί να αναγνωρίζει εικόνες αφού προηγουμένως έχει εκπαιδευτεί κατάλληλα. Η εκπαίδευση ενός νευρωνικού δικτύου γίνεται μέσω κατάλληλων εισόδων (input) και κατάλληλων εξόδων (output) πχ. ως είσοδο να δέχεται **“Η φωτογραφία περιέχει σκύλο ή γάτα;”** και ως έξοδο **“80% γάτα 20% σκύλος”**.

Πως Εκπαιδεύουμε ένα Νευρωνικό Δίκτυο

Ένα νευρωνικό δίκτυο αποτελείται από έναν αριθμό στοιχείων, τους νευρώνες, που κάθε νευρώνας έχει έναν αριθμό σημάτων που έρχονται ως είσοδος σε αυτόν, έχει μερικές καταστάσεις στις οποίες μπορεί να βρεθεί και τέλος έχει μία έξοδο. Κάθε είσοδος έχει μια δική της τιμή βάρους, η οποία υποδηλώνει πόσο στενά είναι συνδεδεμένοι οι δύο νευρώνες που συνδέονται με το βάρος αυτό και η τιμή αυτή κυμαίνεται στο διάστημα $[-1,1]$ και μας δείχνει πόσο δυνατά είναι συνδεδεμένα οι δύο νευρώνες. Όταν ένας νευρώνας ενεργοποιείται, υπολογίζει μια συνάρτηση από όλα τα δεδομένα που έχει και συγκρίνει την τιμή της

συνάρτησης αυτής με μια τιμή κατωφλίου, η οποία είναι χαρακτηριστική για τον νευρώνα αυτόν. Αν η τιμή της συνάρτησης είναι μεγαλύτερη από την τιμή κατωφλίου, τότε ο νευρώνας υπολογίζει την έξοδο, την οποία προωθεί ως είσοδο στον επόμενο νευρώνα. Κατά την διάρκεια της εκπαίδευσης το μόνο πράγμα που αλλάζει είναι οι τιμές των βαρών των συνδέσεων των νευρώνων που αυτό δεν γίνεται πάντα με τον ίδιο τρόπο αλλά εξαρτάται από την μέθοδο την οποία χρησιμοποιούμε. Οι αλλαγές στα βάρη γίνονται με έναν από τους εξής τρεις τρόπους: με εποπτευόμενο τρόπο, με μη-εποπτευόμενο τρόπο και τέλος με αυτο-εποπτευόμενο τρόπο. Η εποπτευόμενη μάθηση συμβαίνει όταν ξεκινάμε με τυχαίες τιμές των βαρών και δίνουμε τις τιμές εισόδων και των στόχων που πρέπει να μάθει το δίκτυο. Κατά την διαδικασία εκπαίδευσης το δίκτυο αλλάζει τις τιμές των βαρών διορθώνοντας αυτές ανάλογο με το σφάλμα που παίρνουμε. Στην μη-εποπτευόμενη εκπαίδευση απλώς δίνουμε την πληροφορία στο δίκτυο, χωρίς να γίνεται κανένας έλεγχος. Στην αυτό-εποπτευόμενη εκπαίδευση το δίκτυο αυτό-ελέγχει τον εαυτό του και διορθώνει τα σφάλματα στα δεδομένα με έναν μηχανισμό ανάδρασης (feedback). Σε όλες τις περιπτώσεις όταν το δίκτυο σταματάει να αλλάζει τις τιμές των βαρών, τότε θεωρούμε ότι η εκπαίδευση έχει επιτευχθεί και αυτό συμβαίνει επειδή το λάθος στην έξοδο γίνεται μηδέν ή είναι πολύ κοντά στο μηδέν.

Από την φύση τους τα νευρωνικά δίκτυα δεν λειτουργούν σειριακά, αλλά με τρόπο που μοιάζει πιο πολύ σε παράλληλο τρόπο λειτουργίας, διότι μια εργασία μοιράζεται στα διάφορα τμήματα του δικτύου, στους νευρώνες κλπ., δηλαδή τα νευρωνικά δίκτυα είναι συστήματα «παράλληλων κατανεμημένων διεργασιών» που έχουν ως αποτέλεσμα να παρέχονται μεγάλες ταχύτητες, διότι είναι σαν να έχουμε ταυτόχρονα πολλούς επεξεργαστές στη διάθεση μας. Ωστόσο, η αρχιτεκτονική των νευρωνικών δικτύων διαφέρει από αυτήν των παράλληλων επεξεργασιών,

επειδή οι απλοί επεξεργαστές των νευρωνικών δικτύων έχουν μεγάλο αριθμό διασυνδέσεων που συνολικά είναι πολύ μεγαλύτερος από τον αριθμό των νευρώνων. Αντιθέτως, στους παράλληλους υπολογιστές, οι επεξεργαστές είναι συνήθως περισσότεροι από ό,τι οι διασυνδέσεις μεταξύ τους και ως προς την πολυπλοκότητα τους ακολουθούν την μηχανή vonNeumann. Στον παρακάτω πίνακα φαίνονται οι διαφορές μεταξύ Νευρωνικών δικτύων και Υπολογιστή.

Νευρωνικά Δίκτυα	Υπολογιστής
Εργάζονται με σύγχρονο τρόπο λειτουργίας	Εργάζονται με ασύγχρονο τρόπο λειτουργίας.
Παράλληλη επεξεργασία	Σειριακή επεξεργασία
Εκπαιδεύονται με παραδείγματα αλλάζοντας τα βάρη των συνδέσεών τους	Προγραμματίζονται με εντολές λογικού χαρακτήρα (if-else)
Η μνήμη, τα δίκτυα και οι μονάδες λειτουργίας συνυπάρχουν	Η μνήμη και επεξεργασία πληροφορίας χωρίζονται
Ανοχή στα σφάλματα	Καμία ανοχή στα σφάλματα
Αυτό-οργάνωση κατά τη διαδικασία της εκπαίδευσης	Εξαρτάται εξ' ολοκλήρου από το προσφερόμενο λογισμικό
Η πληροφορία αποθηκεύεται στα βάρη των συνδέσεων	Η πληροφορία αποθηκεύεται σε addressed memory location
Ο χρόνος ενός κύκλου είναι της τάξης του msec	Ο χρόνος ενός κύκλου είναι της τάξης του nsec

Πλεονεκτήματα των Νευρωνικών Δικτύων

Το επιστημονικό ενδιαφέρον για τα τεχνητά νευρωνικά δίκτυα (ΤΝΔ) προκύπτει κυρίως από τη δυνατότητά τους να επιλύουν δύσκολα και ενδιαφέροντα υπολογιστικά προβλήματα του πραγματικού κόσμου. Η χρήση των ΤΝΔ προσφέρει τις ακόλουθες πολύ χρήσιμες ιδιότητες και δυνατότητες.

1. Αντικατάσταση μαθηματικού μοντέλου. Η αξία των ΤΝΔ έγκειται κυρίως στο γεγονός ότι η χρήση τους δε συνεπάγεται στη δημιουργία κάποιου μοντέλου για το περιβάλλον το οποίο καλούνται να προσομοιώσουν. Όταν η περιγραφή του περιβάλλοντος είναι ιδιαίτερα πολύπλοκη για να περιγραφεί με κάποιο μαθηματικό μοντέλο, τότε η χρήση των ΤΝΔ είναι μια καλή λύση. Κατά την εκπαίδευση δίνονται στο ΤΝΔ πρότυπα εισόδου/εξόδου (δείγματα εισόδων και εξόδων της συνάρτησης που καλείται το ΤΝΔ να προσομοιώσει). Στόχος της διαδικασίας εκπαίδευσης είναι να φτάσει το ΤΝΔ σε μια τέτοια κατάσταση όπου για κάθε πρότυπο εκπαίδευσης, η έξοδός του να ταυτίζεται με την επιθυμητή έξοδο. Έτσι δημιουργείται μια συσχέτιση μεταξύ των δεδομένων εισόδου και εξόδου, χωρίς όμως τη χρήση κάποιου προκαθορισμένου στατιστικού ή άλλου μοντέλου.
2. Προσαρμογή. Τα ΤΝΔ έχουν την ικανότητα να μεταβάλλουν την απόκρισή τους μεταβάλλοντας ελαφρώς τα βάρη των νευρώνων, ανάλογα με τα πρότυπα εισόδου. Συνεπώς, σε περιβάλλοντα τα οποία αλλάζουν διαρκώς, τα ΤΝΔ μπορούν να εφαρμοστούν εφόσον υπόκεινται περιοδικά σε εκπαίδευση με τα επικυρωμένα πρότυπα εισόδου/εξόδου.
3. Δυνατότητα εύκολης υλοποίησης σε Hardware. Η απλή δομή των

νευρώνων και το γεγονός ότι ένα ΤΝΔ αποτελείται από πολλά ίδια στοιχεία (νευρώνες) καθιστά σχετικά απλή την υλοποίηση των ΤΝΔ σε hardware. Με τη χρήση της τεχνολογίας ολοκλήρωσης πολύ μεγάλης κλίμακας (Very Large Scale Integration – VLSI) που είναι ιδιαίτερα διαδομένη για την κατασκευή ηλεκτρονικών πλακετών, μπορούν να δημιουργηθούν εύκολα ταχύτατα ΤΝΔ.

4. Ανεκτικότητα σε σφάλματα. Τα ΤΝΔ που έχουν υλοποιηθεί σε υλικό (hardware) έχουν την ιδιότητα της ανεκτικότητας σε σφάλματα, γιατί η απόδοση του συστήματος μειώνεται ομαλά σε περίπτωση λάθους. Ακόμα και αν καταστραφεί ένας νευρώνας, το ΤΝΔ θα συνεχίσει να λειτουργεί με κάπως μειωμένη απόδοση.
5. Ανοχή στο θόρυβο. Τα ΤΝΔ παρουσιάζουν ανοχή στο θόρυβο υπό την έννοια ότι αν εκπαιδευτούν για την αναγνώριση προτύπων (για παράδειγμα σε εφαρμογές με εικόνες), τότε είναι σε γενικές γραμμές εφικτή η ταξινόμηση από το νευρωνικό ακόμα και όταν η είσοδος έχει υποστεί κάποια αλλοίωση (θόρυβο)

Παραδείγματα νευρωνικών δικτύων στην καθημερινή ζωή

Ακολουθούν μερικά παραδείγματα που τα ΤΝΔ χρησιμοποιούνται σήμερα από διαφορετικές βιομηχανίες:

- **Χρηματοδότηση:** Τα νευρωνικά δίκτυα χρησιμοποιούνται για την πρόβλεψη των συναλλαγματικών ισοτιμιών. Χρησιμοποιούνται επίσης στην τεχνολογία πίσω από τα αυτόματα συστήματα συναλλαγών που χρησιμοποιούνται στη χρηματιστηριακή αγορά.

- **Φάρμακο:** Οι δυνατότητες επεξεργασίας εικόνας των νευρωνικών δικτύων έχουν συμβάλει στην τεχνολογία που βοηθά στην ακριβέστερη ανίχνευση των πρώιμων σταδίων καρκίνου. Ένας τέτοιος τύπος καρκίνου είναι το μελάνωμα, η πιο σοβαρή και θανατηφόρα μορφή του καρκίνου του δέρματος. Ο εντοπισμός του μελανώματος σε προγενέστερα στάδια, προτού εξαπλωθεί, δίνει στους ασθενείς πολλές πιθανότητες ίασης.
- **Καιρός:** Η ικανότητα ανίχνευσης των μεταβολών της ατμόσφαιρας που υποδεικνύουν ένα δυνητικά σοβαρό και επικίνδυνο μετεωρολογικό γεγονός όσο το δυνατόν πιο γρήγορα και με ακρίβεια είναι απαραίτητη για τη διάσωση ζωών. Τα νευρωνικά δίκτυα εμπλέκονται στην επεξεργασία, σε πραγματικό χρόνο, εικόνων, δορυφόρων και ραντάρ που δεν ανιχνεύουν μόνο την πρόωρη δημιουργία τυφώνων και κυκλώνων, αλλά επίσης ανιχνεύουν ξαφνικές μεταβολές στην ταχύτητα και την κατεύθυνση του ανέμου που υποδεικνύουν ίσως κάποιον ένα ανεμοστρόβιλο. Οι ανεμοστρόβιλοι είναι μερικά από τα ισχυρότερα και πιο επικίνδυνα μετεωρολογικά φαινόμενα - συχνά πιο ξαφνικά, καταστροφικά και θανατηφόρα από τους τυφώνες.

Σκοπός της Εργασίας

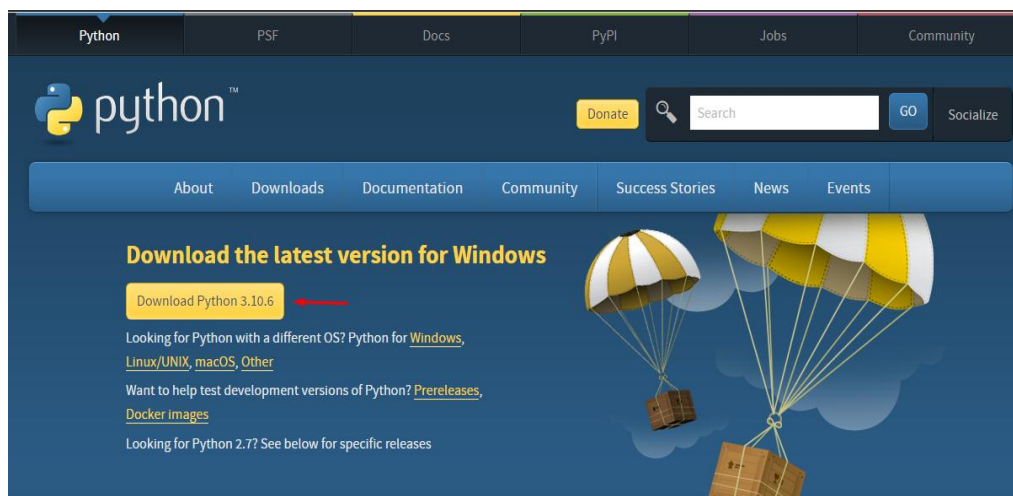
Στη συγκεκριμένη εργασία θα παρουσιαστεί μια υλοποίηση που θα μπορεί ο χρήστης να μάθει πώς θα εγκαταστήσει στον υπολογιστή του τη γλώσσα προγραμματισμού Python. Έπειτα θα μάθει πώς να εγκαταστήσει τις απαραίτητες βιβλιοθήκες pygame και neat και τέλος θα μάθει να «κλωνοποιεί» την αντίστοιχη υλοποίηση από το github.

Εγκατάσταση της Python

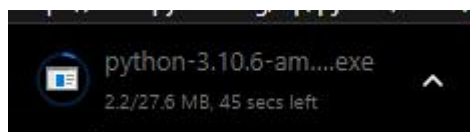
Αρχικά θα πρέπει να μεταβούμε στο url:

<https://www.python.org/downloads/>

Επιλέγουμε να κατεβάσουμε την τελευταία έκδοση (εκεί που δείχνει το κόκκινο βέλος) :

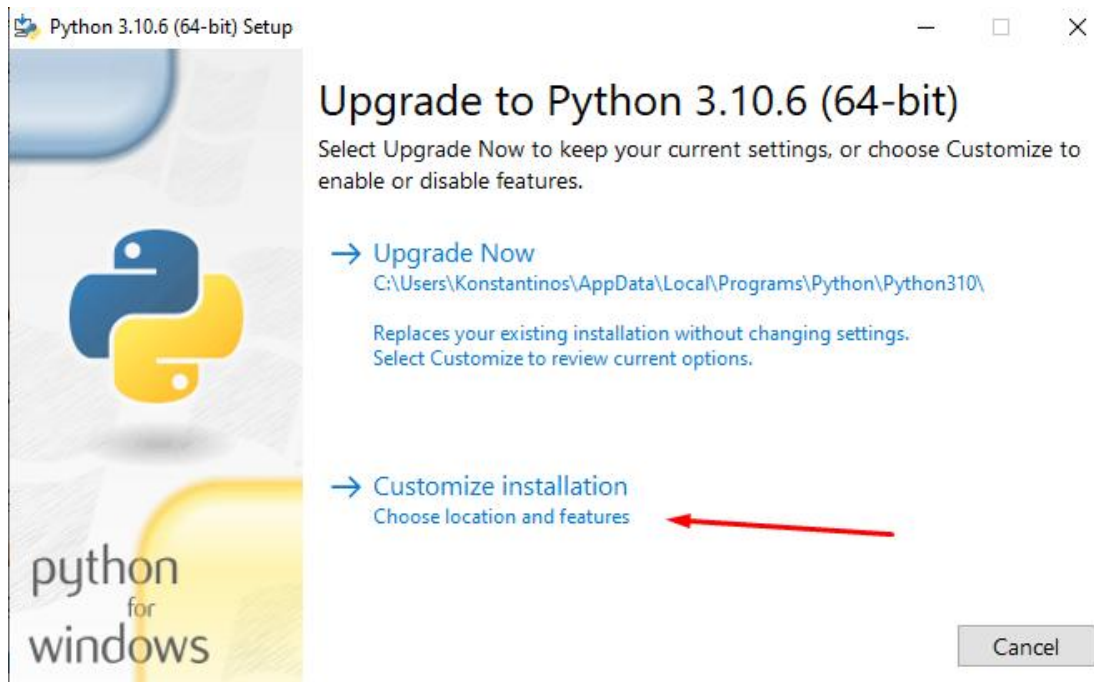


Έπειτα περιμένουμε να ολοκληρωθεί η λήψη μας:

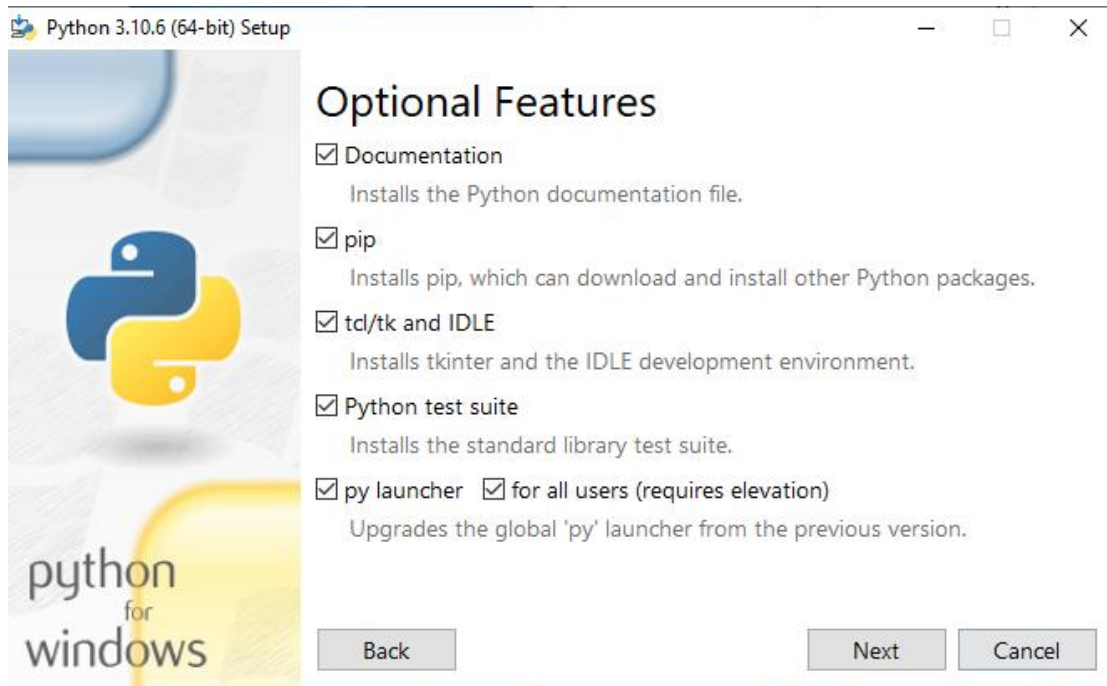


Συνεχίζουμε με την εγκατάσταση της Python ακολουθώντας τα παρακάτω 6 βήματα:

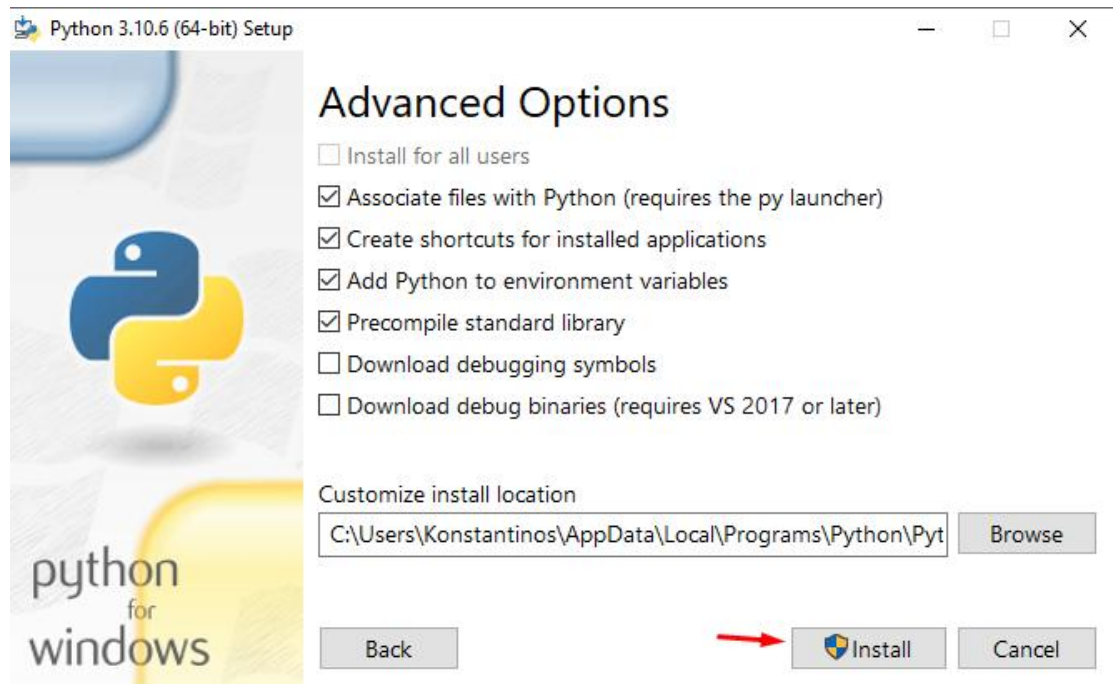
1.



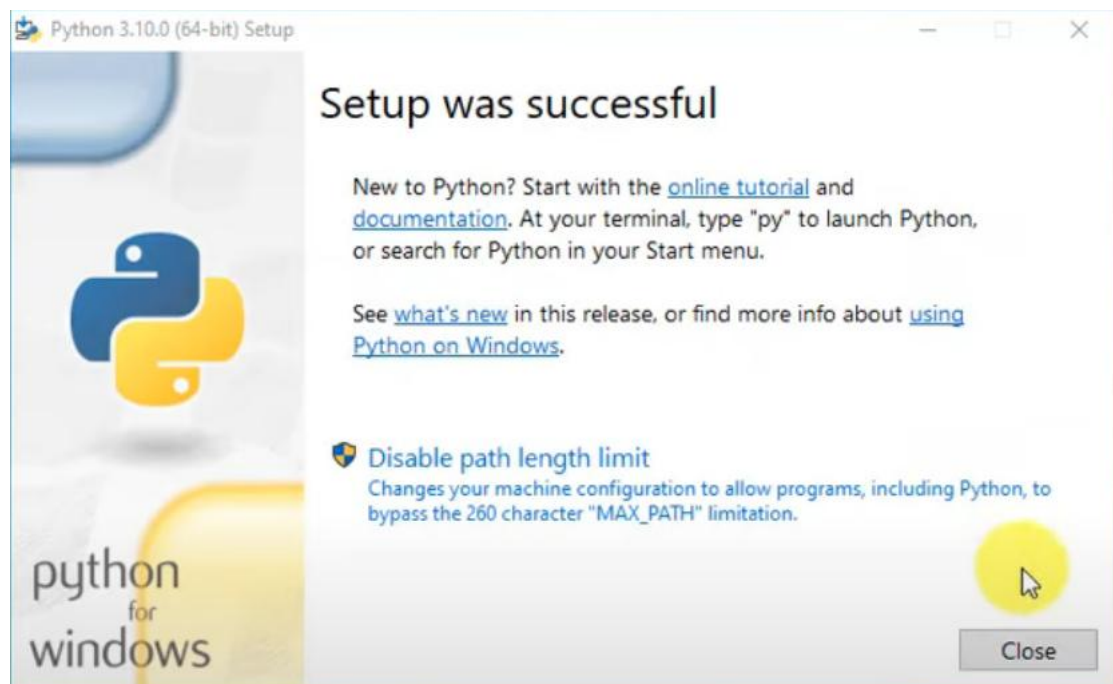
2.



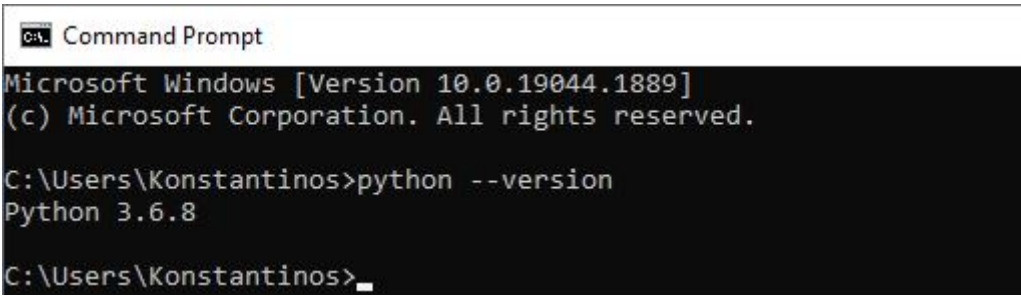
3.



4.



5. Για να επιβεβαιώσουμε ότι έχουμε εγκαταστήσει πλήρως την Python θα πρέπει να εκτελέσουμε την εντολή: **python --version** στον CommandPrompt.

6. 

Έπειτα είμαστε έτοιμοι για να προχωρήσουμε στο επόμενο βήμα που είναι η εγκατάσταση ενός IDE για να τρέξουμε την υλοποίηση μας. Στη συγκεκριμένη υλοποίηση έχω χρησιμοποιήσει το PyCharm, αλλά δεν έχει καμιά διάφορα ως προς το εκτελέσιμο αρχείο αν κάποιος επιθυμεί να το εκτελέσει σε VisualStudioCode.

Εγκατάσταση του PyCharm

Αρχικά θα πρέπει να μεταβούμε στο url:
<https://www.jetbrains.com/pycharm/download/#section=windows>

Επιλέγουμε να κατεβάσουμε την τελευταία έκδοση (εκεί που δείχνει το κόκκινο βέλος):



Version: 2022.2.1
Build: 222.3739.56
17 August 2022

[System requirements](#)
[Installation instructions](#)
[Other versions](#)
[Third-party software](#)

Download PyCharm

[Windows](#) [macOS](#) [Linux](#)

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

[Download](#)

Free 30-day trial available

Community

For pure Python development

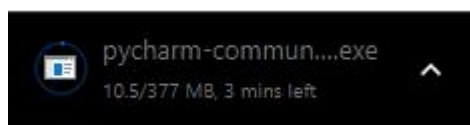
[Download](#)

Free, built on open-source

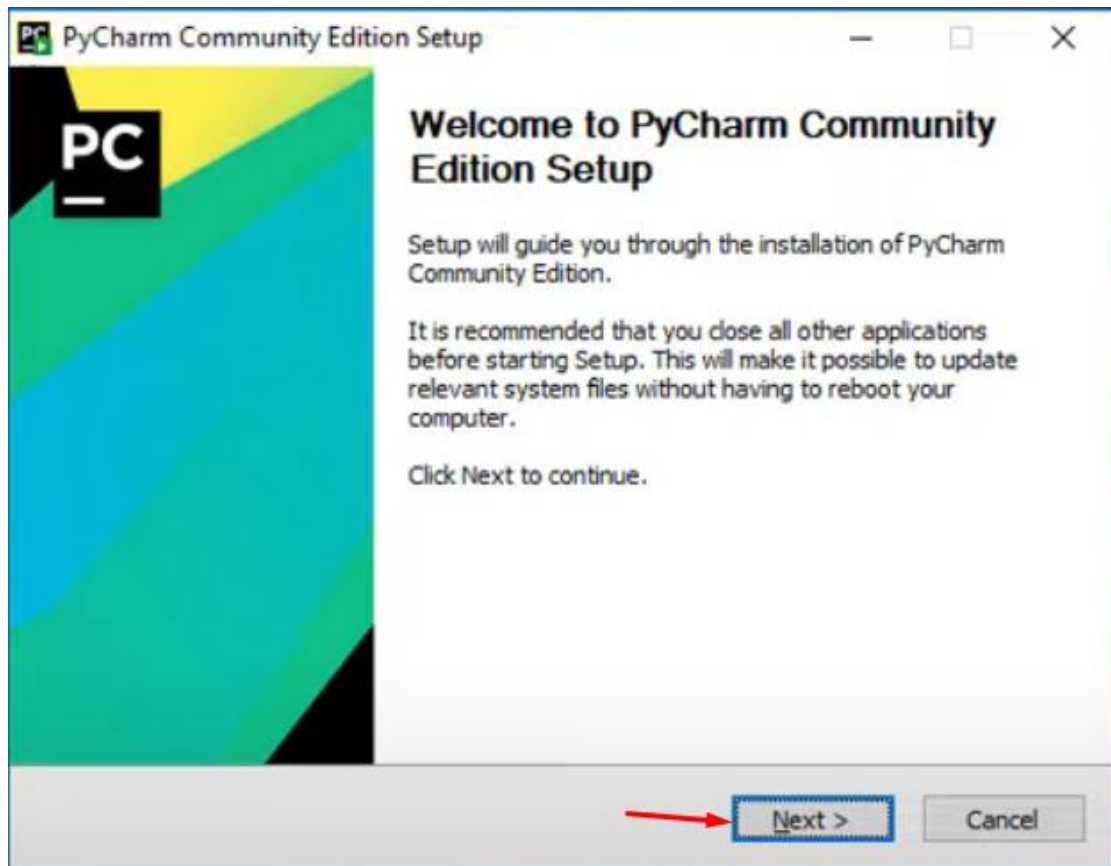


Get the Toolbox App to download PyCharm and its future updates with ease

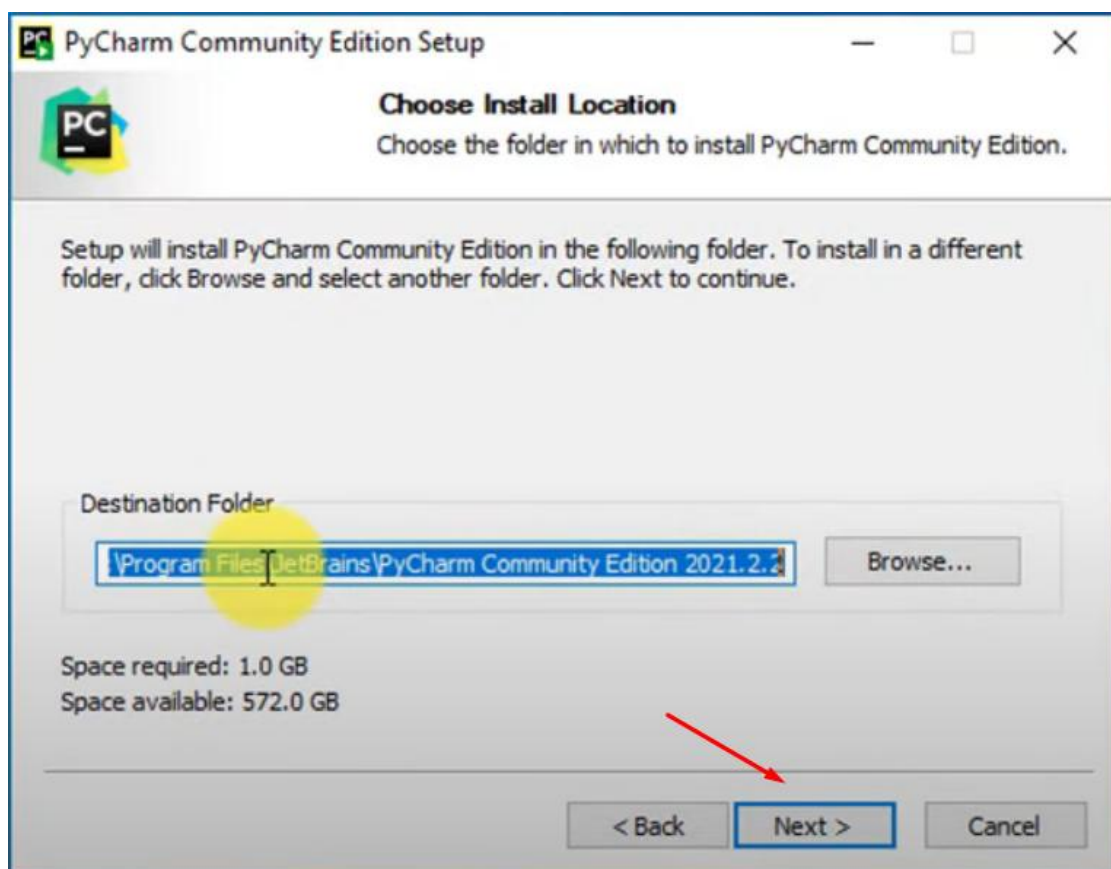
Έπειτα περιμένουμε να ολοκληρωθεί η λήψη μας:



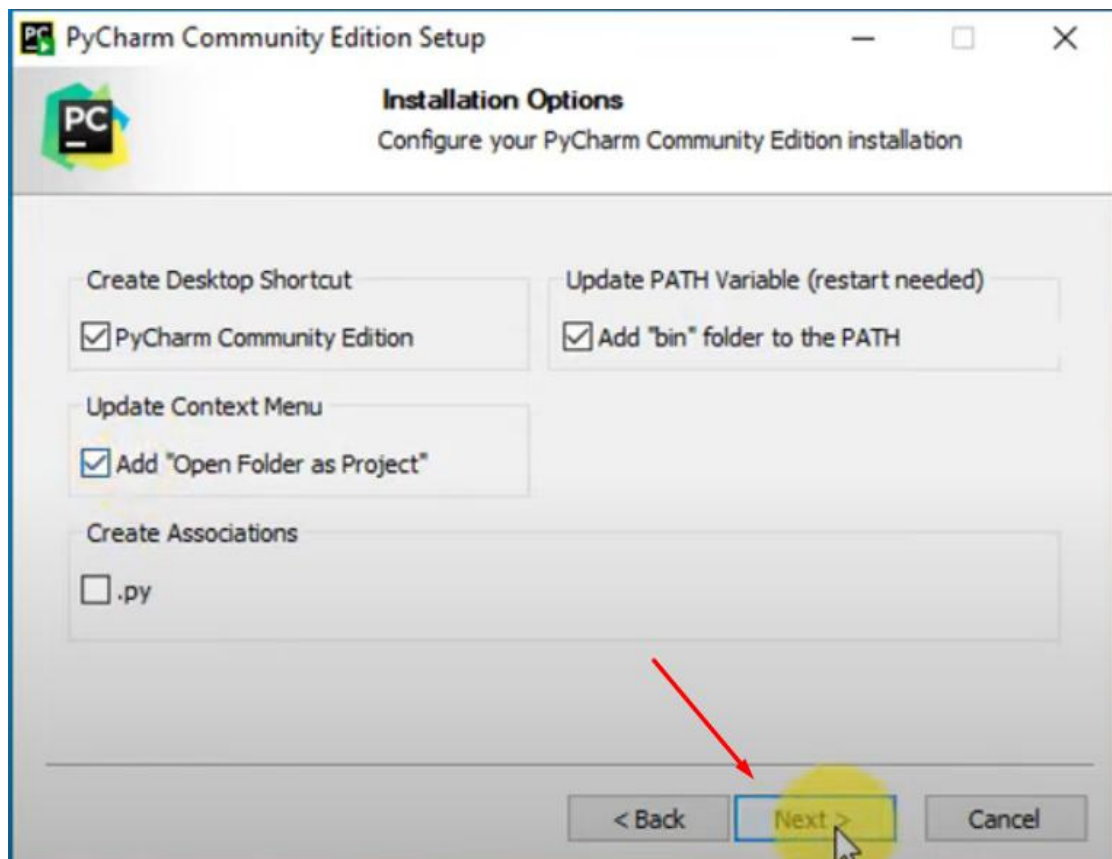
Συνεχίζουμε με την εγκατάσταση του PyCharm ακολουθώντας τα παρακάτω 5 βήματα:



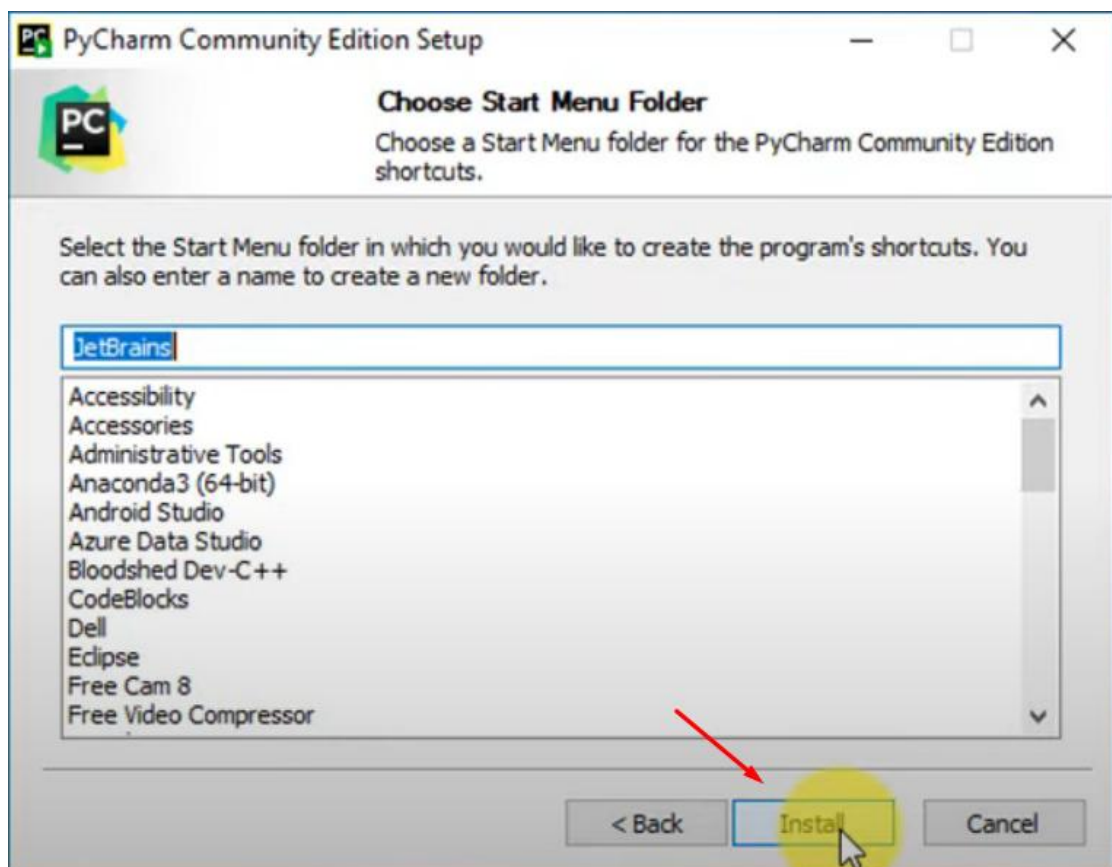
1.



2.

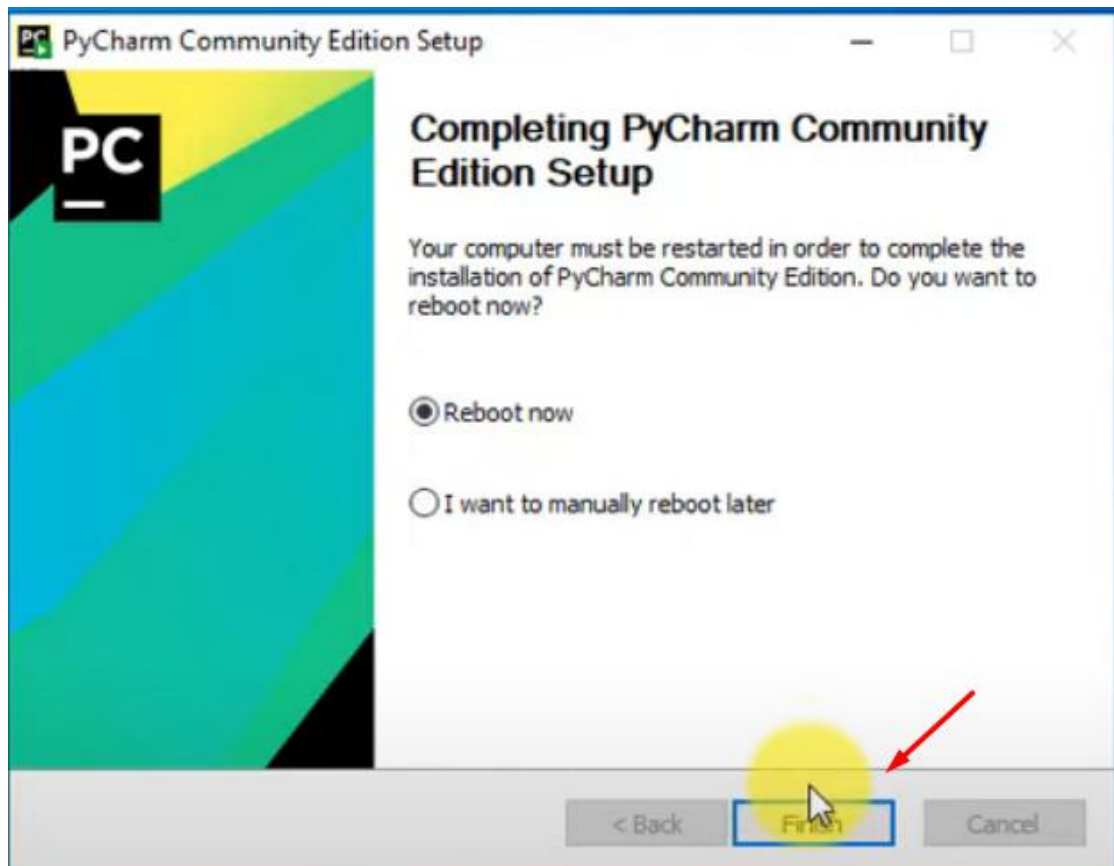


3.



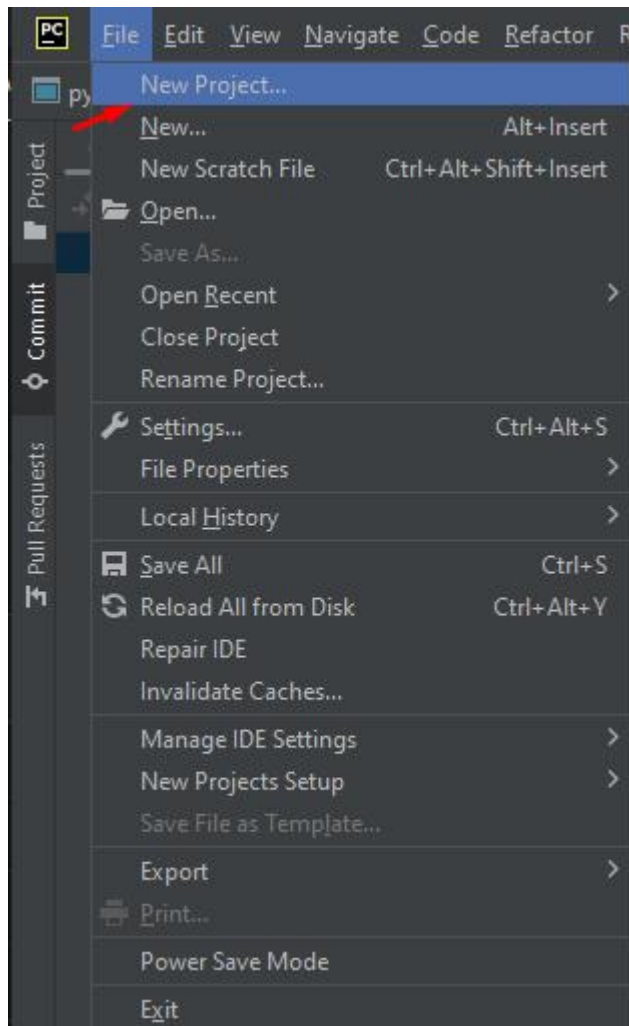
4.

5.

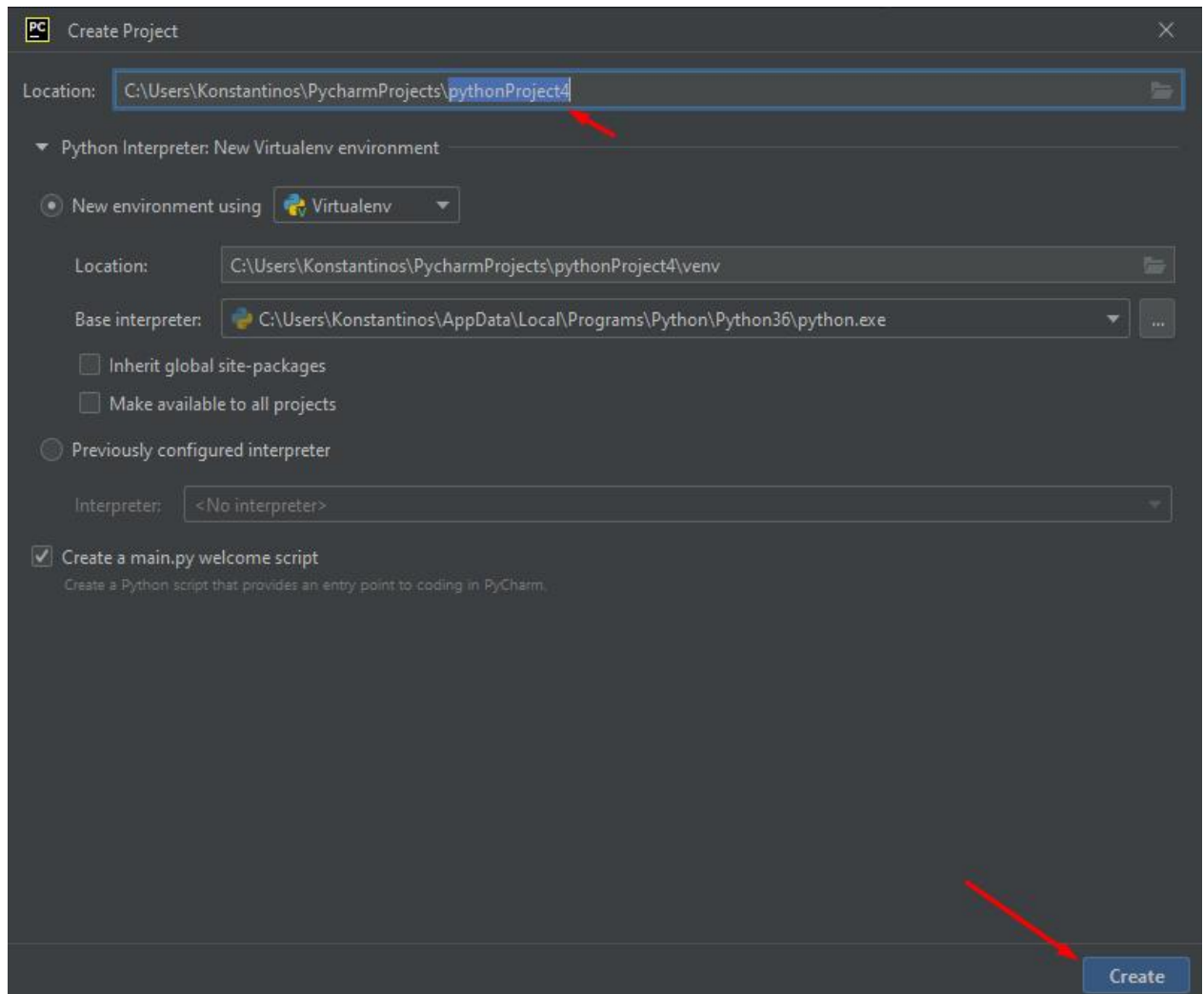


Δημιουργία νέου Project μέσω του PyCharm και Κλωνοποίηση μέσω GitHub

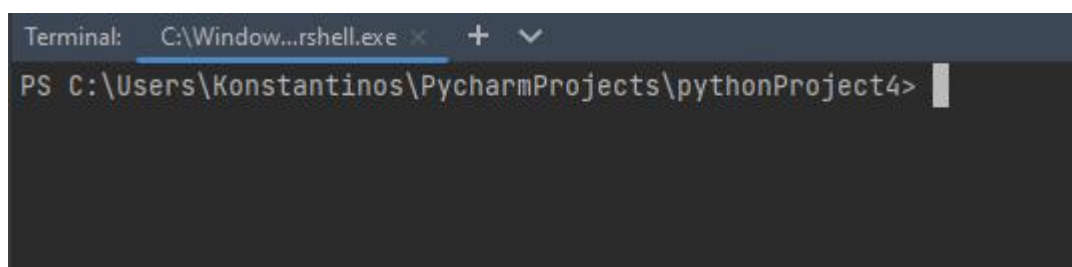
Αφού ανοίξουμε το PyCharm επιλέγουμε **NewProject** όπως στην παρακάτω φωτογραφία.



Έπειτα δίνουμε το όνομα της επιλογής μας στο project και πατάμε **Create**.



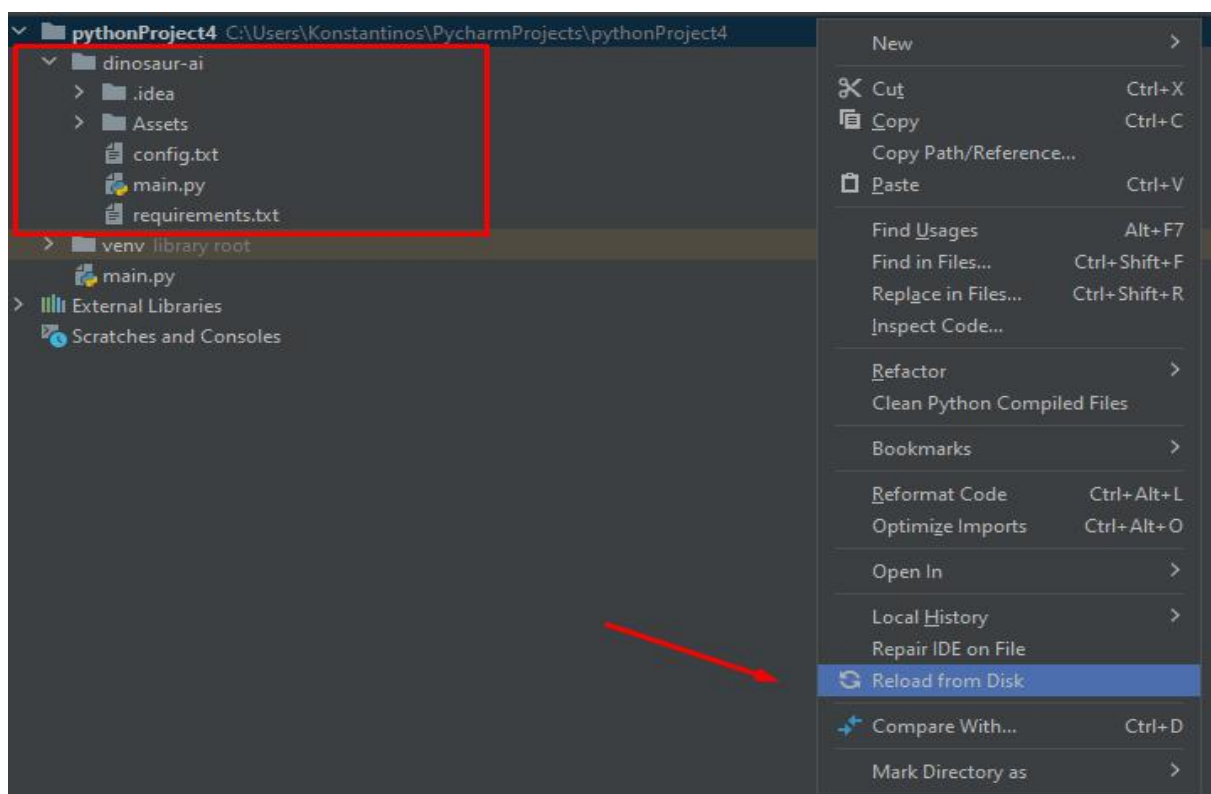
Στη συνέχεια για Windows συστήματα πληκτρολογούμε ALT+F12 για να ανοίξει το terminal όπως στην παρακάτω εικόνα:



Τέλος, μας μένει να «τραβήξουμε» την υλοποίηση μας στον υπολογιστή μας πληκτρολογώντας την εντολή **gitclone** <https://github.com/SylvanasGr/dinosaur-ai.git> :

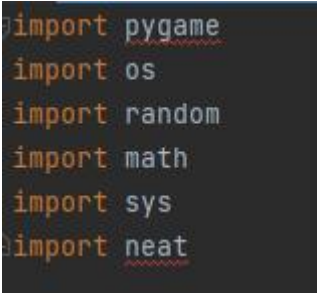
```
Terminal: C:\Window...rshell.exe x + v
PS C:\Users\Konstantinos\PycharmProjects\pythonProject4> git clone https://github.com/SylvanasGr/dinosaur-ai.git
Cloning into 'dinosaur-ai'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (16/16), done.
Receiving objects: 92% (25/27)
Receiving objects: 100% (27/27), 12.33 KiB | 213.00 KiB/s, done.
Resolving deltas: 100% (7/7), done.
PS C:\Users\Konstantinos\PycharmProjects\pythonProject4>
```

Αν έχουμε φτάσει ως εδώ επιτυχώς, θα πρέπει να πατήσουμε δεξί κλικ, την επιλογή **ReloadFromDisk** και θα πρέπει να εμφανίζονται στην οθόνη μας τα παρακάτω αρχεία:

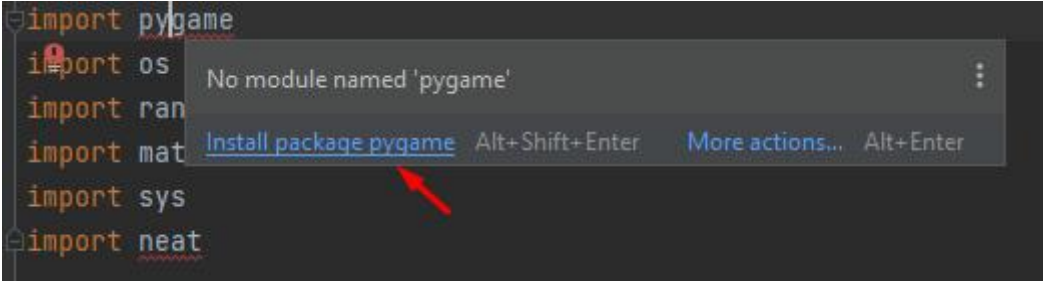


Εγκατάσταση των βιβλιοθηκών pygame και neat

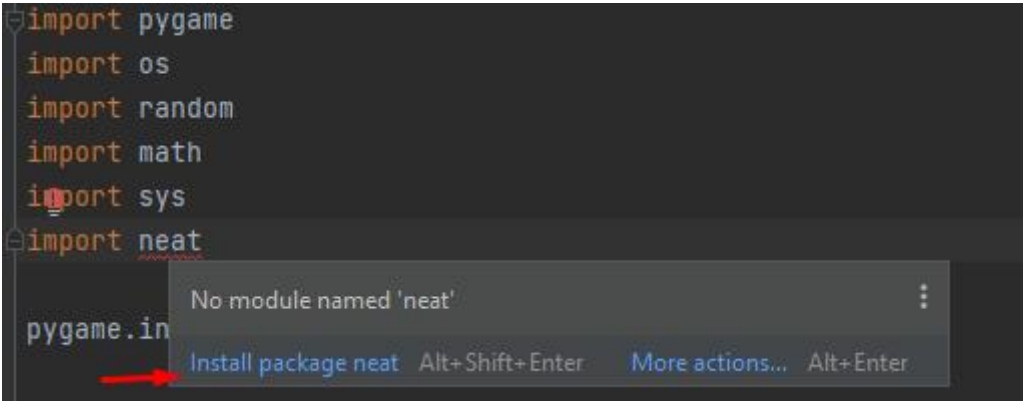
Για την εγκατάσταση των βιβλιοθηκών pygame και neat θα πρέπει να μεταβούμε στο αρχείο main.py και να κάνουμε εγκατάσταση όπως στις παρακάτω εικόνες:

1) 

```
import pygame
import os
import random
import math
import sys
import neat
```

2) 

```
import pygame
import os
import random
import math
import sys
import neat
```

3) 

```
import pygame
import os
import random
import math
import sys
import neat
```

Configurations που αφορούν τον NEAT:

Ο NeuroEvolution of Augmenting Topologies ή αλλιώς NEAT είναι ένας γενετικός αλγόριθμος για τη δημιουργία ΤΝΔ που αναπτύχθηκε από τον Kenneth Stanley το 2002 στο Πανεπιστήμιο του Τέξας στο Ώστιν. Βασίζεται στην εφαρμογή τριών βασικών τεχνικών:

- παρακολούθηση γονιδίων με δείκτες ιστορικού για να επιτραπεί η διασταύρωση μεταξύ τοπολογιών
- εφαρμογή ενδογένεσης για τη διατήρηση των καινοτομιών
- ανάπτυξη τοπολογιών σταδιακά από απλές αρχικές δομές

Περισσότερες πληροφορίες για το NEAT αλγόριθμο υπάρχουν στο επίσημο site τους:

https://neat-python.readthedocs.io/en/latest/neat_overview.html

Στο αρχείο config.txt βρίσκονται τα configuration που χρειάζονται για να τρέξει ο αλγόριθμος NEAT. Τα περισσότερα από αυτά τα έχω αφήσει στις default τιμές όπως είναι ακριβώς από το documentation παραπάνω. Ωστόσο θα περιγράψω τη λειτουργία από τα πιο σημαντικά.

Η βασική ιδέα του αλγορίθμου NEAT είναι ότι στηρίζεται στη θεωρία της εξέλιξης. Δηλαδή, στη συγκεκριμένη υλοποίηση βάζουμε πολλούς δεινόσαυρους να τρέξουν στην πίστα και βάσει πιθανοτήτων μερικοί ή ένας από αυτούς θα κάνουν μεγαλύτερο score από τους υπόλοιπους δεινόσαυρους, όποτε εμείς π.χ. θα βάλουμε 15 δεινόσαυρους και οι 5 από αυτούς με το μεγαλύτερο score θα κλωνοποιήσουν την επόμενη γενιά με μεγαλύτερες πιθανότητες για υψηλότερο score από την προηγούμενη γενιά, δηλαδή το output της γενιάς n θα είναι το input της

γενιάς $n+1$.

1. **fitness_criterion** πρέπει να ισούται με **max**, αυτό σημαίνει ότι θα επιλέγονται οι δεινόσαυροι με τους περισσότερους πόντους.
2. **fitness_threshold** είναι όταν κάποιος δεινόσαυρος πιάσει το αντίστοιχο **score**, τότε τερματίζει το πρόγραμμα μας
3. **pop_size** είναι πόσους δεινοσαύρους θα βγάζουμε σε κάθε generation.
4. **activation_default** πρέπει να ισούται με **tanh** που υπολογίζει το output σύμφωνα με τα input που έχουμε ορίσει.
5. **num_hidden** μπορεί να ισούται με όποια τιμή θέλουμε. Είναι τα **hiddenlayers** μεταξύ **input** και **output**, ωστόσο στη συγκεκριμένη υλοποίηση το άφησα στο 0, σε περίπτωση όμως που κάποιος θέλει να πειραματιστεί με το συγκεκριμένο, συνιστάται να το χρησιμοποιήσει με PythonCuda διότι η ισχύς της κάρτας γραφικών είναι πολύ καλύτερη από την ισχύ του cpu για τέτοιες περιπτώσεις.
6. **num_input** το έχω ορίσει με **2** και δείχνει πόσο ψηλά βρίσκεται ο δεινόσαυρος στον άξονα Y και την απόσταση που έχει ο δεινόσαυρος με τα εμπόδια.
7. **num_output** το έχω ορίσει με **1** και δείχνει αν ο δεινόσαυρος χοροπηδάει ή όχι.
8. **max_stagnation** μας δείχνει πόσους δεινόσαυρους να αφαιρεί μετά από κάθε generation.
9. **species_elitism** όπως και το **elitism** ισούνται με 2 και μας δείχνουν πόσους δεινοσαύρους να «σώζει» για κλωνοποίηση για τις επόμενες γενιές με βάση την καλύτερη απόδοση που είχαν.

Λεπτομέρειες για το Παιχνίδι και Βίντεο επεξήγησης της υλοποίησης:

Στην παρόν παιχνίδι ο δεινόσαυρος τρέχει και πρέπει να ξεπερνάει τα εμπόδια. Όσο αυξάνεται ο χρόνος που μένει ζωντανός ο δεινόσαυρος, αυξάνεται και η ταχύτητα του παιχνιδιού. Τέλος το παιχνίδι τερματίζει μόνο όταν ο δεινόσαυρος έρθει σε επαφή με κάποιο εμπόδιο.

**Για να δείτε το βίντεο παρακαλώ ανοίξτε το βίντεο που σας έστειλα
“επεξήση παιχνιδιού”.**