ΔΙΑΔΙΚΑΣΙΕΣ ΠΟΥ ΚΑΛΟΥΝ ΑΛΛΕΣ ΔΙΑΔΙΚΑΣΙΕΣ

Στα προηγούμενα παρατηρήσαμε ότι για την επίλυση ενός σύνθετου προβλήματος μπορούμε να δημιουργήσουμε δικές μας διαδικασίες. Με άλλα λόγια δικές μας σύνθετες εντολές.

Εξηγήσαμε ότι αυτό γίνεται προκειμένου να αποφύγουμε σύνθετες προγραμματιστικές δομές (όπως οι εντολές εμφωλευμένων (nested) "Επανάλαβε"), που οδηγούν σε δυσνόητο κώδικα. Πάλι όμως, οι ανάγκες ενός σύνθετου προβλήματος μπορεί να μας παρασύρουν στη συγγραφή πολύπλοκων και ως εκ τούτου δυσνόητων διαδικασιών. Προκειμένου να μην υποπέσουμε σε αυτά τα λάθη όπως και στον κίνδυνο να γράψουμε δυσνόητα και δύσχρηστα προγράμματα, υπενθυμίζουμε τη φιλοσοφία της λογικής της Ανάλυσης Προβλημάτων (Σχ.1.1 - σελ. 177 -178 σχολικού βιβλίου).

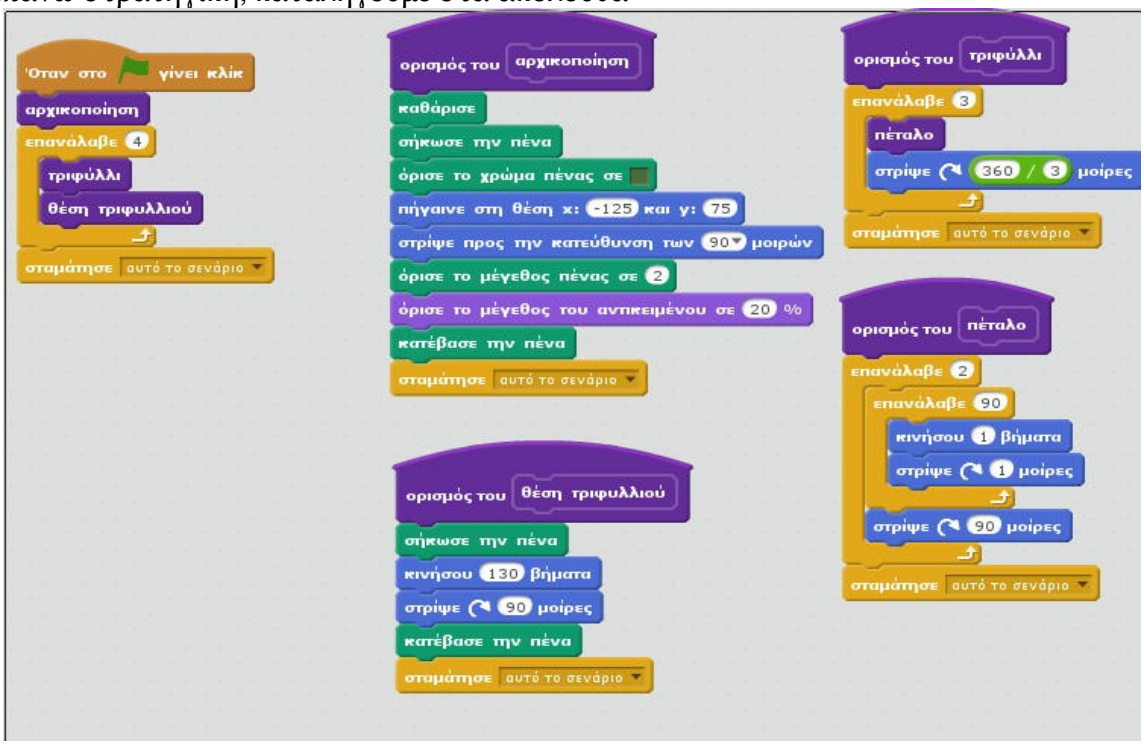
Ανάλυση Προβλήματος (Μέθοδος Διαίρει και Βασίλευε / Divide and Conquer): Για να επιλύσουμε ένα σύνθετο πρόβλημα, είναι αναγκαίο να το αναλύσουμε σε απλούστερα προβλήματα.

Εφαρμογή της συγκεκριμένης φιλοσοφίας στο προγραμματιστικό περιβάλλον Scratch 2.0, στον προγραμματισμό με Logo και με οποιαδήποτε άλλη γλώσσα προγραμματισμού γενικότερα, είναι με την ανάλυση του προβλήματος με πολλές διαδικασίες, που επιλύουν στοιχειώδη – απλά μέρη του συνόλου. Δηλαδή ακολουθούμε μια προσέγγιση από το μέρος προς το όλον ή όπως συνηθίζεται να λέγεται στην Πληροφορική από κάτω προς τα πάνω / bottom – up).

Για παράδειγμα, για να λύσουμε το πιο σύνθετο πρόβλημα **ενός κήπου με τέσσερα τριφύλλια σε διάταξη τετραγώνου**, θα ξεκινήσουμε την επίλυση από μέρος του προβλήματος, μέχρι να επιλύσουμε το συνολικό πρόβλημα:

1. Σχεδίαση πετάλου
2. Σχεδίαση τριφυλλιού
3. Σχεδίαση τεσσάρων τριφυλλιών.

Χρήσιμη είναι μια διαδικασία για το συμμάζεμα των εντολών που έχουν να κάνουν με την αρχικοποίηση και την προετοιμασία της σκηνής για τη σχεδίαση όπως και μία διαδικασία για την μετακίνηση της πένας και τη σχεδίαση κάθε τριφυλλιού. Τέλος, θα χρειαστεί και από μία διαδικασία για το σχεδιασμό του πετάλου και του τριφυλλιού αντίστοιχα. Χρησιμοποιώντας την παραπάνω στρατηγική, καταλήγουμε στα ακόλουθα:

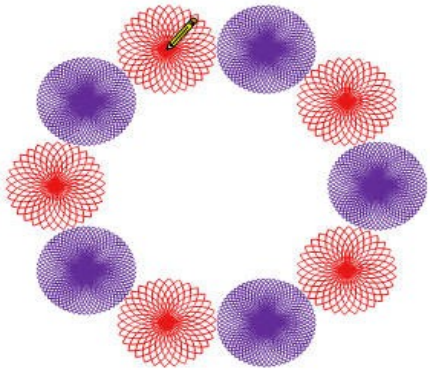


Με αυτή τη δομή έχουμε πετύχει να διατηρούμε **απλό** και **κατανοητό** τον κυρίως κώδικα (το σενάριο που εκτελείται “Όταν στο πράσινο σημαιάκι γίνει κλικ”), ενώ στις επιμέρους διαδικασίες έχουμε **αναλύσει την πολυπλοκότητα** του προβλήματός μας, που χτίσαμε **βαθμωτά.** Δηλαδή γράφοντας αρχικά τις πιο απλές διαδικασίες που λύνουν τα στοιχειώδη υποπροβλήματα και καταλλήγοντας στις πιο σύνθετες διαδικασίες που ενδεχομένως καλούν τις απλούστερες.

Τέλος, ένα ακόμη σημαντικό προσόν σε αυτή την αρχιτεκτονική γραφής προγράμματος, είναι η **επεκτασιμότητα.** Μπορούμε να συνεχίσουμε να γράφουμε πιο σύνθετα προγράμματα, χτίζοντας και γράφοντας πιο σύνθετες διαδικασίες.

ΔΡΑΣΤΗΡΙΟΤΗΤΕΣ - ΕΡΓΑΣΙΕΣ

1. Να γράψετε ένα πρόγραμμα που χρησιμοποιώντας τις έτοιμες διαδικασίες “πέταλο” και “τριαντάφυλλο” αλλά και μία “εξηντάφυλλο” που θα δημιουργήσετε που σχεδιάζει ένα στεφάνι με 10 λουλούδια (εναλλάξ τριαντάφυλλα εξηντάφυλλα). Θεωρήστε ότι τα κέντρα των λουλουδιών απέχουν 80 pixels το ένα από το άλλο.



2. Βλ. 1η δραστηριότητα από τη σελίδα 192 του σχολικού βιβλίου. Φτιάξτε ένα πρόγραμμα που σχεδιάζει ένα χωριό (πχ 5 σπίτια σε μία γραμμή που απέχουν 50 pixels το ένα από το άλλο), αφού σχεδιάσετε σπίτια με διαδικασίες "τρίγωνο","τετράγωνο" που σχεδιάζουν τη σκεπή και το κτήριο αντίστοιχα.