Πως λύνεται ένα πρόβλημα.

Όπως έχουμε ήδη αναφέρει, τα βήματα για την παραγωγή λογισμικού είναι:

1. Κατανόηση προβλήματος
2. Επίλυση του προβλήματος
3. Λογικός έλεγχος της λύσης (αν υπάρχουν λάθη πήγαινε στο 1.)
4. Κωδικοποίηση
5. Μετάφραση & Ορθογραφικός έλεγχος (αν υπάρχουν λάθη πήγαινε στο 4)
6. Λογικός έλεγχος (αν υπάρχουν λάθη πήγαινε στο 2)
7. Παραγωγή (εκτέλεση του προγράμματος σε πραγματικές συνθήκες)

Η εξειδίκευση των τριών πρώτων βημάτων που αποτελούν και το αντικείμενο του μαθήματος είναι:

1. Ανάλυση του προβλήματος
2. Σύνθεση της λύσης
3. Έλεγχος της λύσης
4. Εξέταση οριακών συνθηκών
5. Οριστικοποίηση της λύσης.

Ανάλυση του προβλήματος

Κατά την ανάλυση του προβλήματος γίνονται τα παρακάτω βήματα κατά σειρά:

- Καταγραφή δοκιμαστικών δεδομένων που προκύπτουν αβίαστα από την εκφώνηση του προβλήματος.
- Καταγραφή των αποτελεσμάτων που πρέπει να προκύψουν από την επεξεργασία αυτών των δεδομένων.
- Ανάλυση της ανθρώπινης επεξεργασίας σε λειτουργίες υπολογιστή.

Το τελευταίο βήμα είναι και το δυσκολότερο, δεδομένου ότι ο υπολογιστής έχει τις γνωστές τρεις δυνατότητες: πρόσθεση, σύγκριση, μεταφορά. Αυτό σημαίνει ότι κάθε μορφής επεξεργασία θα πρέπει να καταγραφεί σαν λειτουργία υπολογιστή. Όλα τα παραπάνω αποτελούν την αρχή της τεκμηρίωσης, που θα συνοδεύει κάθε φάση.

Σύνθεση

Σε αυτή την φάση αναζητείται κατ' αρχήν η βασική επεξεργασία που θα περιλαμβάνεται στην επαναληπτική διαδικασία. Θυμίζουμε ότι, ένα πρόβλημα προκειμένου να είναι κατάλληλο για λυθεί από υπολογιστή πρέπει να έχει κάποιο από τα παρακάτω χαρακτηριστικά ή συνδυασμό τους: επαναλαμβανόμενες λειτουργίες, μεγάλο όγκο δεδομένων, πολύπλοκους υπολογισμούς.
Έτσι τα παρακάτω βήματα θα πρέπει να είναι:

- Καταγραφή της βασικής επεξεργασίας του αλγόριθμου. Αυτή θα είναι σύμφωνη με τις δυνατότητες του υπολογιστή.
- Συμπλήρωση με υποστηρικτικές εντολές, προκειμένου η βασική επεξεργασία να είναι ολοκληρωμένη και πλήρης.
- Καθορισμός της επαναληπτικής διαδικασίας με την επιλογή της κατάλληλης δομής σύμφωνα με τις δεδομένες συνθήκες.
- Τεχνικός έλεγχος ο οποίος θα διαπιστώσει, ποιες εντολές δεν μπορούν να εκτελεστούν λόγω έλλειψης προϋποθέσεων. Συνήθως αφορά σε συγκρίσεις μεταβλητών που δεν έχουν περιεχόμενο, ή σε αριθμητικές πράξεις με πεδία χωρίς αρχικές τιμές. Ακόμα θα διαπιστώσει αν ο αλγόριθμος έχει είσοδο, επεξεργασία, έξοδο.

Κάθε παραδοχή, όπως για παράδειγμα, αρχικές τιμές και τιμές τερματισμού πρέπει να συμπληρώνουν την τεκμηρίωση αυτής της φάσης.

Έλεγχος λειτουργίας.

Σε αυτή την φάση ο αλγόριθμος ελέγχεται με τα δοκιμαστικά δεδομένα που είχαν χρησιμοποιηθεί και για τον σχεδιασμό του.

- Εκτελείται ο αλγόριθμος, με "δοκιμή στο γραφείο". Τα παραγόμενα αποτελέσματα συγκρίνονται με τα αναμενόμενα. Κάθε απόκλιση σημαίνει ότι υπάρχει λογικό λάθος.
- ΠΡΟΣΟΧΗ: Κάθε φορά που διαπιστώνεται λογικό λάθος και γίνεται η διόρθωσή του, ο έλεγχος πρέπει να αρχίζει από την αρχή.
- Κάθε παραδοχή που γίνεται, αποτελεί στοιχείο της τεκμηρίωσης του φακέλου του προγράμματος.

Εξέταση ακραίων συνθηκών.

Πριν παραδοθεί ο αλγόριθμος για τα επόμενα βήματα, πρέπει να εξεταστεί η συμπεριφορά του σε ακραίες περιπτώσεις δεδομένων. Έτσι σε αυτή η φάση:

- Γίνεται αλλαγή των δεδομένων προκειμένου να εξεταστούν όλες δυνατές περιπτώσεις.
- Κάθε νέα ομάδα δεδομένων καταγράφεται μαζί με τα αποτελέσματα της.
- Ελέγχεται η λειτουργία του αλγόριθμου. Αν χρειαστεί να γίνουν αλλαγές στις λειτουργίες του, τότε ΥΠΟΧΡΕΩΤΙΚΑ αυτές οι αλλαγές θα πρέπει να εξεταστούν και με τα δεδομένα που έχουν χρησιμοποιηθεί μέχρι στιγμής.
- Κάθε παρατήρηση καταγράφεται και αποτελεί στοιχείο της τεκμηρίωσης.

Οριστικοποίηση της λύσης.

- Ένας τελευταίος έλεγχος αποδεικνύει ότι έχουν ληφθεί υπ' όψιν όλες οι πιθανές περιπτώσεις.
- Η λύση οριστικοποιείται.

Εφαρμογή

Δίνονται αριθμοί από το πληκτρολόγιο. Να σχεδιαστεί ο αλγόριθμος που εντοπίζει τον μεγαλύτερο από αυτούς και την θέση του.
Σημείωση: η περίπτωση πολλαπλού μέγιστου, ισότητας όλων των αριθμών κ.ά θα εξεταστούν στη φάση των ακραίων συνθηκών.

Ανάλυση του προβλήματος

Αν x είναι η μεταβλητή εισόδου, max η βοηθητική μεταβλητή που αποθηκεύει τον μέγιστο και θ η μεταβλητή που κρατά την θέση του αριθμού, ο παρακάτω πίνακας μπορεί να είναι μία περίπτωση που λύνει το πρόβλημα.

x	Max	θ
2	7	4
4		
-1		
7		
5		
0		
τ		

Η μεταβλητή max προφανώς παίρνει τιμή μετά από σύγκριση του x και του max. Άρα η βασική επεξεργασία είναι:

```
Αν  x>max
    τότε max ← x
Τέλος Αν
```

Επειδή όμως πρέπει να αποθηκεύεται και η θέση του x χρειάζεται ένας μετρητής που θα μετρά την σειρά της τιμής που έχει διαβαστεί και αποθηκεύεται στο max.
Επομένως ο πίνακας γίνεται:

x	Max	θ	i
2	7	4	1
4			2
-1			3
7			4
5			5
0			6
τ			

Και η εντολή γίνεται:

```
Αν  x>max
    τότε max ← x
         θ ← i
Τέλος Αν
```

Εφ' όσον αυτή εντολή επαναλαμβάνεται, προφανώς θα πρέπει να εκτελείται με νέα δεδομένα. Αυτά δεν μπορεί να είναι άλλα από νέα τιμή για το x και νέα τιμή για το i. Επειδή το x είναι εξωτερική μεταβλητή παίρνει τιμή με την εντολή Διάβασε, ενώ το i σαν εσωτερική μεταβλητή παίρνει τιμή με πράξη, στην προκειμένη περίπτωση πρόσθεση.
Η επεξεργασία είναι τώρα:

```
Αν  x>max
    τότε max ← x
         θ ← i
Τέλος Αν
i ← i + 1
Διάβασε x
```

Σημείωση: Παρακάτω θα δούμε πως η θέση του i ← i + 1, επηρεάζει την αρχική τιμή της μεταβλητής.
Μετά και από αυτό το βήμα, μπορούμε να ορίσουμε την επαναληπτική διαδικασία με αρκετή βεβαιότητα ότι δεν έχουμε παραλείψει τίποτα.
Η επεξεργασία θα μπορεί να είναι τώρα:

```
Όσο x≠"τ" επανάλαβε
    Αν  x>max
        τότε  max ← x
              θ ← i
    Τέλος Αν
    i ← i + 1
    Διάβασε x
```

```
Τέλος Επανάληψης
```

Ο τεχνικός έλεγχος δίνει τις εξής παρατηρήσεις:

- Η συνθήκη x≠"τ" δεν μπορεί να ελεγχθεί γιατί το x δεν έχει τιμή.
- Η συνθήκη x > max δεν μπορεί να ελεγχθεί γιατί ούτε το x ούτε το max έχουν τιμή.
- Η εντολή i ← i + 1 δεν μπορεί να εκτελεστεί γιατί το i δεν έχει αρχική τιμή.
- Ο αλγόριθμος δεν έχει έξοδο.

Ο αλγόριθμος γίνεται τώρα:

```
i ← 1
Διάβασε x
max ← x
Όσο x≠"τ" επανάλαβε
    Αν  x>max
        τότε  max ← x
              θ ← i
    Τέλος Αν
    i ← i + 1
    Διάβασε x
Τέλος Επανάληψης
```

Προσοχή: Η αρχικοποίηση της max με την τιμή του πρώτου x που διαβάζεται είναι η καλλίτερη επιλογή.

το επόμενο βήμα συμπληρώνει τον αλγόριθμο με την έξοδο:

```
i ← 1
Διάβασε x
max ← x
Όσο x≠"τ" επανάλαβε
    Αν  x>max
        τότε  max ← x
              θ ← i
    Τέλος Αν
    i ← i + 1
    Διάβασε x
Τέλος Επανάληψης
Εμφάνισε max, θ
```

σε αυτό το σημείο ο αλγόριθμος είναι έτοιμος για έλεγχο. Για τον έλεγχο θα χρησιμοποιήσουμε τα ίδια δεδομένα, ώστε να ελεγχθούν τα αποτελέσματα.

x	Max	θ	i	Παρατηρήσεις
~~2~~	~~2~~		~~1~~	πριν αρχίσει η επανάληψη
4	4	2	2	όταν το x=4 το I=2
-1				
7				
5				
0				
τ				

Ο έλεγχος σταματά γιατί παρατηρούμε ότι ενώ το max παίρνει την τιμή 2 το θ δεν παίρνει την τιμή 1, όπως έπρεπε αλλά πρώτη τιμή του είναι το 2. Αυτό συμβαίνει γιατί δεν παίρνει αρχική τιμή (εκτός επανάληψης).
επομένως ο αλγόριθμος διορθώνεται και γίνεται:

```
i ← 1
Διάβασε x
max ← x
θ ← i
Όσο x≠"τ" επανάλαβε
    Αν  x>max
        τότε   max ← x
               θ ← i
    Τέλος Αν
    i ← i + 1
    Διάβασε x
Τέλος Επανάληψης
Εμφάνισε max, θ
```

Τώρα ο αλγόριθμος παρουσιάζει και το χαρακτηριστικό της τυποποίησης. Δηλαδή οι μεταβλητές max και θ παίρνουν με την ίδια τιμή, είτε είναι αρχική τιμή είτε αλλαγή λόγω επεξεργασίας.
ο έλεγχος επαναλαμβάνεται από την αρχή.

x	Max	θ	i	Παρατηρήσεις
2	2		1	πριν αρχίσει η επανάληψη
4	4	2	2	όταν το x=4 το I=2
-1	7	4	3	όταν το x=7 το I=4
7			4	
5			5	
0			6	
τ				

Αυτός ο έλεγχος οριστικοποιεί τον αλγόριθμο.
Σαν ακραίες περιπτώσεις μπορούμε να θεωρήσουμε την ύπαρξη δύο ή περισσότερων "μεγίστων" στα δεδομένα.
Σε αυτή την περίπτωση πρέπει να είναι σαφές αν ο αλγόριθμος δίνει σωστό αποτέλεσμα και ποιου μέγιστου δίνει την θέση.
Μια δοκιμή θα δώσει τις απαντήσεις.
Ιδιαίτερη συζήτηση πρέπει να γίνει για την περίπτωση του πολλαπλού μέγιστου.

Άσκηση:
Επειδή οι μέχρι στιγμής "γνώσεις" δεν επιτρέπουν την διατύπωση αλγόριθμου, για την περίπτωση του πολλαπλού μέγιστού, ζητείται από τους φοιτητές να διατυπώσουν την λύση με την φραστική μέθοδο.