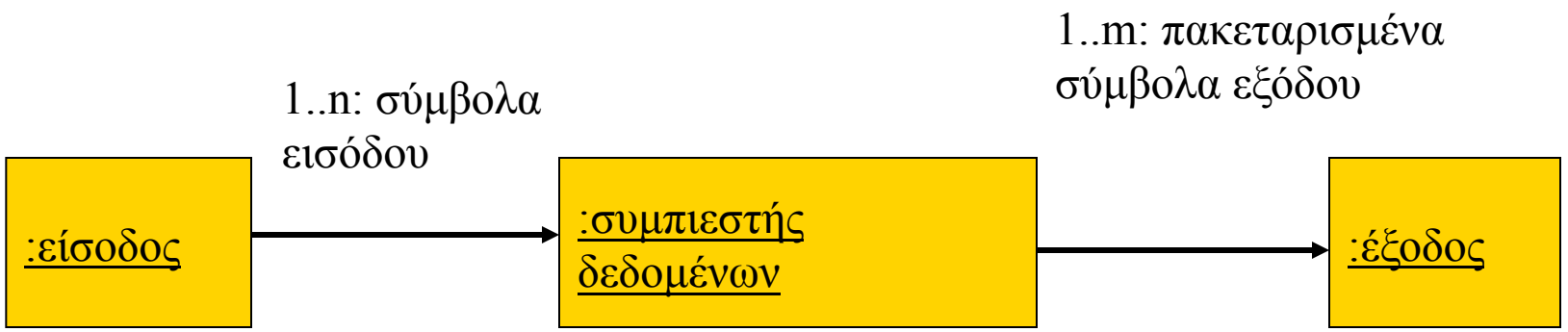
CPUs

- Παράδειγμα: συμπιεστής δεδομένων.

Στόχοι

- Συμπιέζει τα δεδομένα που μεταφέρονται πέρα από την σειριακή γραμμή.
  - Λαμβάνει σύμβολα εισόδου μεγέθους byte.
  - Παράγει σύμβολα εξόδου που πακετάρονται σε bytes.
- Μόνο εδώ θα χτίσει την ενότητα λογισμικού.

Διάγραμμα συνεργασίας για το συμπιεστή

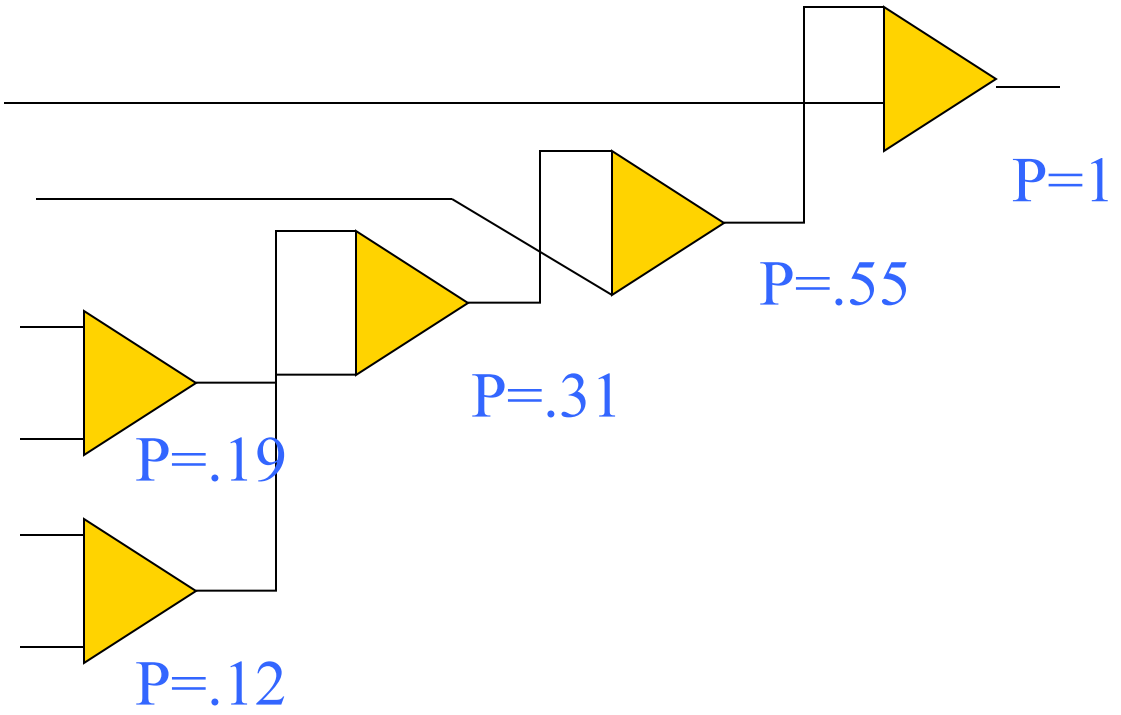


Κωδικοποίηση Huffman

- Πρόωρος στατιστικός αλγόριθμος συμπίεσης κειμένων.
- Επιλέξτε ανομοιόμορφου μεγέθους κώδικες.
  - Χρησιμοποιεί μικρότερους κώδικες για περισσότερο κοινά σύμβολα.
  - Χρησιμοποιεί μεγαλύτερους κώδικες για λιγότερο κοινά σύμβολα.
- Για να επιτρέψει την αποκωδικοποίηση, οι κώδικες πρέπει να έχουν μοναδικά προθέματα.
  - Κανένας κώδικας δεν μπορεί να είναι ένα πρόθεμα ενός πιο μακροχρόνιου έγκυρου κώδικα.

Παράδειγμα Huffman

character	P
a	.45
b	.24
c	.11
d	.08
e	.07
f	.05



Παράδειγμα κώδικα Huffman

- Διαβάστε τον κώδικα από κάτω προς τα πάνω (from root to leaves):

a	1
b	01
c	0000
d	0001
e	0010
f	0011

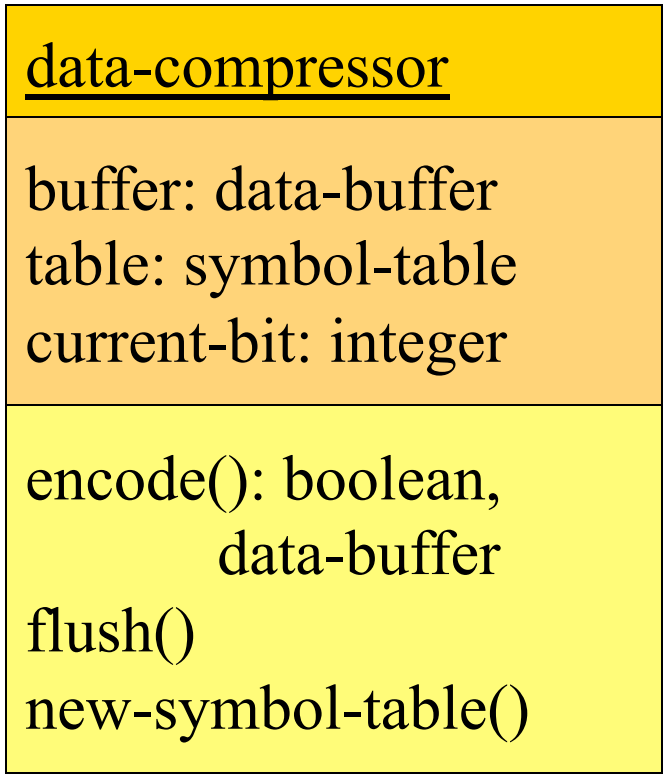
Πίνακας απαιτήσεων κωδικοποιητή Huffman

όνομα	ενότητα συμπίεσης δεδομένων
σκοπός	ενότητα κώδικα για τη συμπίεση Huffman
είσοδοι	πίνακας κωδικοποίησης, μη κωδικοποιημένες είσοδοι μεγέθους byte
έξοδοι	πακεταρισμένα σύμβολα συμπίεσης εξόδου
λειτουργίες	κωδικοποίηση Huffman
απόδοση	γρήγορη
κόστος παραγωγής	ΑΠΡΟΣΔΙΟΡΙΣΤΟ
ισχύς	ΑΠΡΟΣΔΙΟΡΙΣΤΟ
φυσικό μέγεθος/βάρος	ΑΠΡΟΣΔΙΟΡΙΣΤΟ

Χτίζοντας μια προδιαγραφή

- Το διάγραμμα συνεργασίας παρουσιάζει μόνο κατάσταση εισόδου-εξόδου.
- Ένα πραγματικό σύστημα πρέπει:
  - Να δέχεται ένα κωδικοποιημένο πίνακα.
  - Να επιτρέπει ένα σύστημα να επαναρυθμίζεται, το οποίο flushes τον buffer συμπίεσης.

Κλάση data-compressor (συμπιεστής δεδομένων)



Συμπεριφορά της κλάσης data-compressor

- κωδικοποίηση: παίρνει μια είσοδο ενός byte, παράγει συσκευασμένα κωδικοποιημένα σύμβολα και μια ένδειξη Boolean εάν ο buffer είναι πλήρης
- πίνακας νέου συμβόλου: εγκαθιστά νέο πίνακα συμβόλων στο αντικείμενο, πετάει τον παλαιό πίνακα
- flush: επιστρέφει την τρέχουσα κατάσταση του buffer, συμπεριλαμβανομένου του αριθμού έγκυρων bits στον buffer

Βοηθητικές κλάσεις

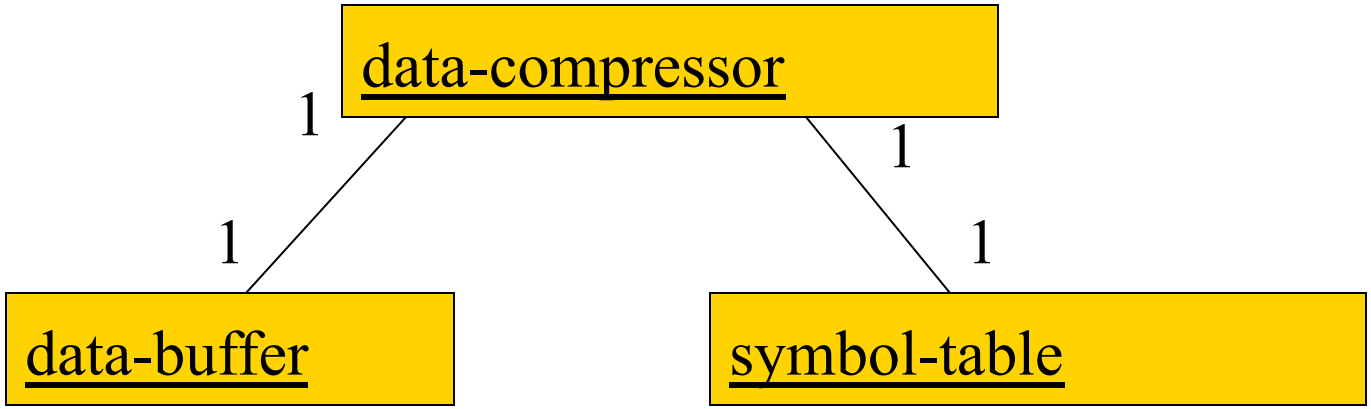
data-buffer
databuf[databuflen] : character len : integer
insert() length() : integer

symbol-table
symbols[nsymbols] : data-buffer len : integer
value() : symbol load()

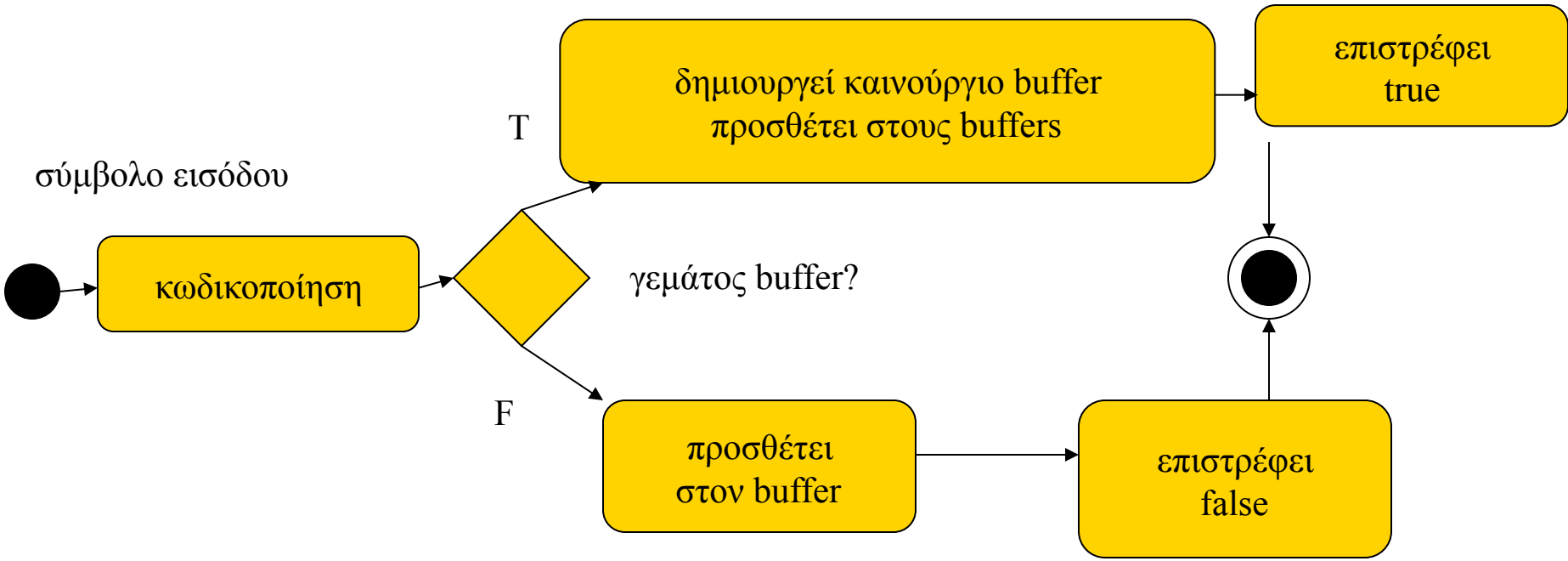
Ρόλος βοηθητικών κλάσεων

- Η κλάση data-buffer κρατάει και τα πακεταρισμένα και τα απακετάριστα σύμβολα.
  - Ο μεγαλύτερος κώδικας Huffman για εισόδους των 8-bit είναι 256 bits.
- Ο πίνακας συμβόλων συντάσσει την κωδικοποιημένη έκδοση κάθε συμβόλου
  - Η λειτουργία load() βάζει δεδομένα σε ένα νέο πίνακα συμβόλων.

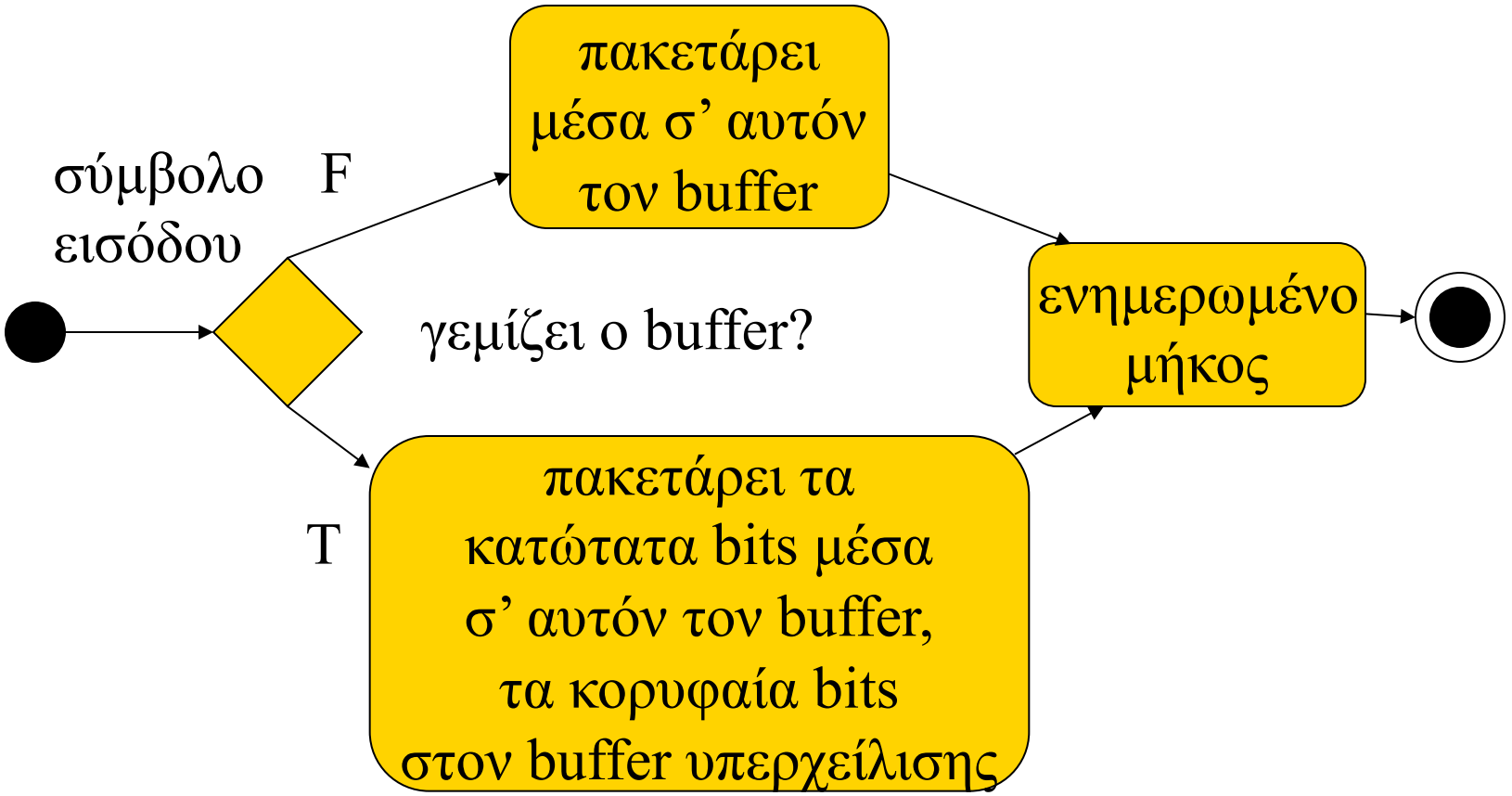
Σχέσεις κλάσεων



Συμπεριφορά κωδικοποίησης



Συμπεριφορά ενθέτων



Σχεδίαση προγράμματος

- Σε μια αντικειμενοστρεφή γλώσσα, μπορούμε να απεικονίσουμε την προδιαγραφή της UML στον κώδικα πιο άμεσα.
- Σε μια μη- αντικειμενοστρεφή γλώσσα, πρέπει είτε:
  - να προσθέσουμε τον κώδικα για να παρέχει αντικειμενοστρεφή χαρακτηριστικά γνωρίσματα
  - να την αποκλείσουμε από τη δομή προδιαγραφών

Κλάσεις στη C++

```
Class data_buffer {
    char databuf[databuflen];
    int len;
    int length_in_chars() { return len/bitsperbyte; }
public:
    void insert(data_buffer,data_buffer&);
    int length() { return len; }
    int length_in_bytes() { return (int)ceil(len/8.0); }
    int initialize();
    ...
```

Κλάσεις στη C++, συν.

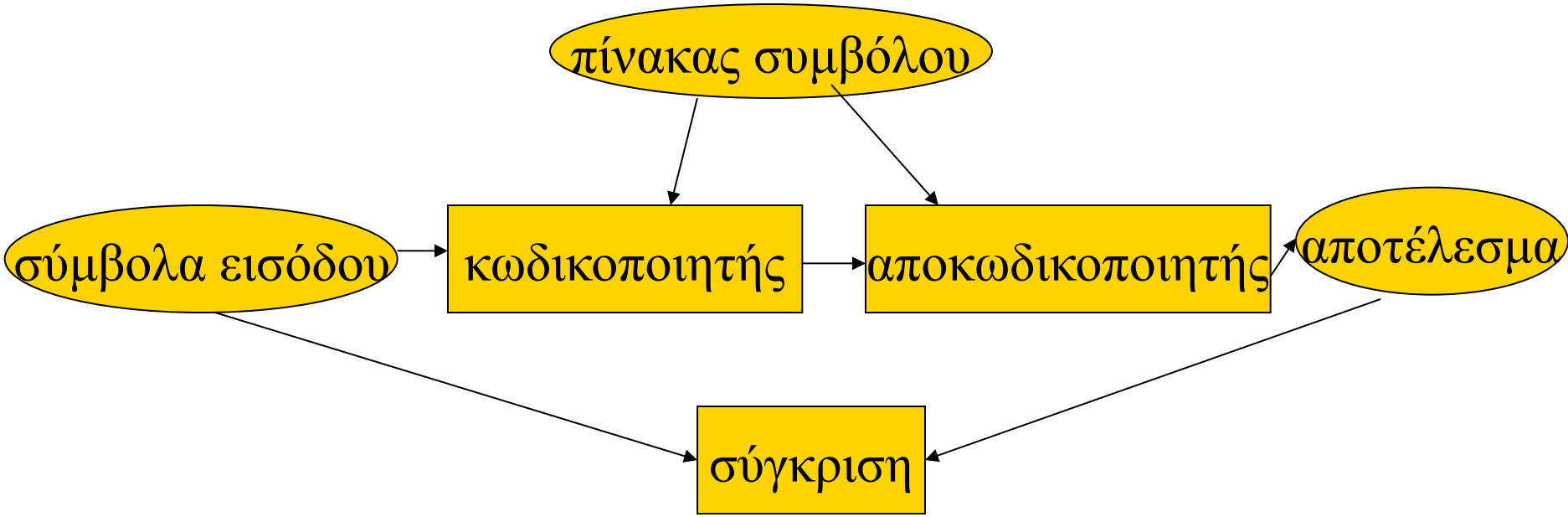
```
class data_compressor {
    data_buffer buffer;
    int current_bit;
    symbol_table table;
public:
    boolean encode(char,data_buffer&);
    void new_symbol_table(symbol_table);
    int flush(data_buffer&);
    data_compressor();
    ~data_compressor();
    }
```

Κώδικας C

```
struct data_compressor_struct {
    data_buffer buffer;
    int current_bit;
    sym_table table;
}
typedef struct data_compressor_struct data_compressor,
    *data_compressor_ptr;
boolean data_compressor_encode(data_compressor_ptr
    mycmptrs, char isymbol, data_buffer *fullbuf) ...
```

Δοκιμή

- Δοκιμάζουμε κωδικοποιώντας, μετά αποκωδικοποιώντας:



Δοκιμές επιθεώρησης κώδικα

- Εξετάζουμε τον κώδικα για πιθανά προβλήματα:
  - Μπορούμε να τρέξουμε το προηγούμενο τέλος του πίνακα συμβόλων;
  - Τι συμβαίνει όταν το επόμενο σύμβολο δεν γεμίσει τον buffer; Γεμίζει;
  - Δουλεύουν κατάλληλα τα πολύ μεγάλα κωδικοποιημένα σύμβολα; Τα πολύ μικρά σύμβολα;
  - Η λειτουργία flush() δουλεύει κατάλληλα;