Το παρόν README, αφορά την Άσκηση 1, της Εργασίας 1.

class Student:

Η κλάση Student αναφέρεται στον μαθητή.

Το private section αποτελείται από το όνομα του (name), από το που βρίσκεται αυτή τη στιγμή (at) και 2 integers (fl~floor) & (cl~class) που αναφέρονται στον όροφο και την τάξη που βρίσκεται αντίστοιχα.

Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *Student(string nam, int f, int c)*
  Είναι ο constructor του Student που δέχεται ως ορίσματα το όνομα του, τον όροφο και την τάξη που ανήκει. Στο σώμα του, αρχικοποιούνται τα μέλη του Student βάσει των ορισμάτων και το member “at” παίρνει την τιμή “OFF” που σημαίνει ότι ο μαθητής βρίσκεται ακόμα εκτός σχολείου. Τέλος εκτυπώνονται τα στοιχεία του.
- *~Student()*
  Είναι ο destructor του Student που εμφανίζει μόνο ότι ο μαθητής με όνομα name καταστράφηκε.
- *void print()*
  Η συνάρτηση print εκτυπώνει μόνο το όνομα του μαθητή (χωρίς endl).
- *void getF_CL(int &f, int &c)*
  Η συνάρτηση getF_CL επιστρέφει το floor και το class που βρίσκεται ο μαθητής.
- *string getAT()*
  Η συνάρτηση getAT επιστρέφει το που βρίσκεται ο μαθητής κατά την κλήση της συνάρτησης (δηλαδή school yard, stairs, class κλπ).
- *void setAT(string place)*
  Η συνάρτηση setAT επεξεργάζεται το member “at” της κλάσης και το μεταβάλει ανάλογα με το που όρισμα που της δόθηκε (αν ο μαθητής μπει στα stairs τότε το όρισμα ειναι “stairs”).

class Teacher:

Η κλάση Teacher αναφέρεται στον καθηγητή.

Το private section αποτελείται από το όνομα του (name) και από 3 integers (fl~floor), (cl~class) & (f~flag) που αναφέρονται στον όροφο, την τάξη και το αν βρίσκεται μέσα στην τάξη (f=0 ~ εκτός τάξης / f=1 ~ εντός τάξης).

Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *Teacher(string nam, int f, int c, int FLAG=0)*
  Είναι ο constructor του Teacher που δέχεται ως ορίσματα το όνομα του, τον όροφο και την τάξη που ανήκει καθώς και το αν βρίσκεται μέσα ή οχι στην τάξη (από default η τιμή είναι 0, δηλαδή βρίσκεται εκτός). Στο σώμα του, αρχικοποιούνται τα μέλη του Teacher βάσει των ορισμάτων. Τέλος εκτυπώνονται τα στοιχεία του.
- *~Teacher()*
  Είναι ο destructor του Teacher που εμφανίζει μόνο ότι ο καθηγητής με όνομα name καταστράφηκε.
- *void print()*
  Η συνάρτηση print εκτυπώνει μόνο το όνομα του καθηγητή (χωρίς endl).
- *void getF_CL(int &f, int &c)*
  Η συνάρτηση getF_CL επιστρέφει το floor και το class που βρίσκεται ο καθηγητής.
- *void setFLAG()*
  Η συνάρτηση setFLAG δίνει την τιμή “1” στο member “flag” της κλάσης.
- *int getFLAG()*
  Η συνάρτηση getFLAG επιστρέφει την τιμή του member “flag” της κλάσης.

class Class:

Η κλάση Class αναφέρεται στην τάξη.

Το private section αποτελείται από τον integer (ppl~people) που αναφέρεται στο πόσα άτομα βρίσκονται μέσα στην τάξη, τον integer (capacity) που αναφέρεται στο πόσα άτομα χωράνε στην τάξη, το Student** list που είναι ένας πίνακας που περιέχει pointers που δείχνουν σε Student και ουσιαστικά περιέχει του μαθητές που βρίσκονται μέσα στην τάξη και το Teacher* teacher που είναι ένας pointer που δείχνει σε Teacher και ουσιαστικά είναι ο καθηγητής της τάξης.

Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *Class(int cap)*
  Είναι ο constructor της τάξης και δέχεται ως όρισμα το πόσα άτομα χωράνε συνολικά σε αυτή (capacity).
- *~Class()*
- Είναι ο destructor της τάξης, στο σώμα του υπάρχει σε σχόλιο το μήνυμα ότι καταστράφηκε η τάξη και μόνο (ο destructor της τάξης καλείται όταν τερματίζει το πρόγραμμα, για αυτό δεν έκανα και delete τους πίνακες, που θα ήταν και πιο σωστό).
- *void print()*
  Η συνάρτηση print εκτυπώνει τον καθηγητή, αν υπάρχει, και όποιον από του μαθητές έχουν προλάβει να μπουν στην τάξη τους.
- *void getCL(int &i, int &j, int &flag)*
  Η συνάρτηση getCL επιστρέφει μέσω των 3 ορισμάτων της, το πόσα άτομα βρίσκονται μέσα (i = cl), το πλήθος των μαθητών που χωράνε στην τάξη (j = capacity) και το αν βρίσκεται μέσα στην τάξη ο καθηγητής ή όχι.
- *void enter(Student* st)*
  Η συνάρτηση enter, εισάγει έναν μαθητή μέσα στην τάξη του, προσθέτοντάς τον στο πίνακα “list” και αυξάνοντας το πλήθος των μαθητών που βρίσκονται μέσα στην τάξη, αν γίνεται να μπει (δηλαδή αν υπάρχει χωρητικότητα και αν δεν είναι μέσα ο καθηγητής).
- *void place(Teacher* t, int f, int c)*
  Η συνάρτηση place, εισάγει έναν καθηγητή μέσα στην τάξη του (αναθέτει τον pointer του καθηγητή στο member teacher της κλάσης)

class Corridor:

Η κλάση Corridor αναφέρεται στον διάδρομο.

Το private section αποτελείται από τα ίδια members της κλάσης Class, μη τη διαφορά ότι δεν υπάρχει ο pointer που δείχνει σε Teacher.

Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *Corridor(int cap)*
  Το όρισμα αρχικοποιεί το "capacity" member της κλάσης, δηλαδή τα συνολικά άτομα που χωράνε στον διάδρομο.
- *~Corridor()*
- *void print()*
- *void getCORR(int &i, int &j)*
- *void enter(Student* st)*
  - Όλες οι παραπάνω συναρτήσεις, έχουν τις ίδιες λειτουργίες με όσα αναφέρθηκαν για την κλάση Class (getCL~getCORR με την διαφορά ότι δεν υπάρχει το flag για το αν βρίσκεται ο καθηγητής ή όχι).
- *void leave(Student* st)*
  Η συνάρτηση leave, καλείται όταν ο μαθητής μπορεί να πάει στο επόμενο επίπεδο (πχ αν από corridor πάει class, καλείται τόσο η enter της κλασης Class, όσο και η leave της κλασης Corridor, κάτι που ισχύει και για τις παρακάτω κλάσεις).

class Floor:

Η κλάση Floor αναφέρεται σε έναν όροφο.

Το private section αποτελείται από 6 pointers που δείχνουν σε Class (θα μπορούσε να είχε γίνει και πίνακας, αλλά το υλοποίησα έτσι) και από έναν pointer που δείχνει σε Corridor.

Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *Floor(int capCL, int capCORR)*
  Ο constructor του Floor καλεί τον constructor του Class και του Corridor, επομένως τα ορίσματα capCL & capCORR είναι το "capacity" των κλάσεων Class και Corridor αντίστοιχα.
- *~Floor()*
- *void print()*
- *void enter(Student* st, int i, int j, int &q)*
- *void place(Teacher* t, int f, int c)*
  - Όλες οι παραπάνω συναρτήσεις, έχουν τις ίδιες λειτουργίες με όσα αναφέρθηκαν για τις παραπάνω κλάσεις (για place βλ. class Class).

class Stairs:

Η κλάση Stairs αναφέρεται στο κλιμακοστάσιο.
Το private section αποτελείται από τα ίδια members με το Corridor.
Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *Stairs(int cap);*
  Το όρισμα αρχικοποιεί το “capacity” member της κλάσης, δηλαδή τα συνολικά άτομα που χωράνε στο κλιμακοστάσιο.
- *~Stairs()*
- *void print()*
- *void getSTAIRS(int &i, int &j)*
- *void enter(Student* st)*
- *void leave()*
    - Όλες οι παραπάνω συναρτήσεις, έχουν τις ίδιες λειτουργίες με όσα αναφέρθηκαν για τις παραπάνω κλάσεις.

class Schoolyard:

Η κλάση Schoolyard αναφέρεται στο προαύλιο.
Το private section αποτελείται από τα ίδια members με το Corridor/Stairs.
Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *Schoolyard(int cap)*
- Το όρισμα αρχικοποιεί το “capacity” member της κλάσης, δηλαδή τα συνολικά άτομα που χωράνε στο προαύλιο.
- *~Schoolyard()*
- *void print()*
- *void getYARD(int &i, int &j)*
- *void enter(Student* st)*
- *void leave(Student* st)*
    - Όλες οι παραπάνω συναρτήσεις, έχουν τις ίδιες λειτουργίες με όσα αναφέρθηκαν για τις παραπάνω κλάσεις.

class School:

Η κλάση School αναφέρεται σε όλο το σχολείο.

Το private section αποτελείται από 1 pointer που δείχνει σε Schoolyard , από έναν pointer που δείχνει σε Stairs, από 3 pointers που δείχνουν σε Floor (θα μπορούσε να είναι και πίνακας), 2 integers (STppl~StudentPeople) & (TCHRppl~TeacherPeople) που είναι μεταβλητές που μετρούν πόσοι μαθητές και καθηγητές έχουν μπει στο σχολείο και από 2 πίνακες που περιέχουν pointers που δείχνουν σε Student ο ένας και σε Teacher ο άλλος.

Στο public section βρίσκονται μόνο οι διάφορες συναρτήσεις που επεξεργάζονται τα παραπάνω members.

- *School(int CL, int CORR, int ST, int YARD);*
  Ο constructor του School καλεί τους constructors των κλάσεων Schoolyard (YARD), Stairs (ST), Corridor (CORR) & Class (CL), περνώντας τα αντίστοιχα ορίσματα που δηλώνουν το capacity των αντικειμένων.
- *~School()*
- *void print()*
- *void enter(Student* st)*
- *void place(Teacher* t, int f, int c)*
- *void leave()*
  - Όλες οι παραπάνω συναρτήσεις, έχουν τις ίδιες λειτουργίες με όσα αναφέρθηκαν για τις παραπάνω κλάσεις.

ΛΕΙΤΟΥΡΓΙΑ MAIN ΚΑΙ ΡΟΗ ΠΡΟΓΡΑΜΜΑΤΟΣ.

Στη main δημιουργούνται οι πίνακες που περιέχουν pointers που δείχνουν σε Student και σε Teacher αντίστοιχα. Αφού γίνει memory allocation, διαβάζω από ένα αρχείο ("studentsFULL.txt") τα δεδομένα των μαθητών και από το αρχείο ("teachersFULL.txt") τα δεδομένα των καθηγητών.

>>!SOS! Τα δεδομένα είναι σε μορφή Name-FloorNum-ClassNum (πχ Mesrine-2-3 που σημαίνει ότι το όνομα του είναι Mesrine, ο όροφος στον οποίο βρίσκεται η τάξη του είναι ο 2ος, και η τάξη του είναι η 3η).

Στη main διαβάζω γραμμή-γραμμή τα δεδομένα και κάθε γραμμή είναι ένα string. Θέτω ως delimiter το σύμβολο “-” για να μπορώ να ξεχωρίζω τα διάφορα δεδομένα μεταξύ τους. Σε ένα πιο δυναμικό περιβάλλον θα μπορούσα να χρησιμοποιώ cin, ζητώντας από το χρήστη να μου δίνει 1-1 τα στοιχεία των μαθητών/καθηγητών.

Αφού διαβαστούν, δημιουργηθούν και αποθηκευτούν τα αντικείμενα, δημιουργείται ένα “σχολικό κτίριο”, [School school(CL,CORR,ST,Y);] καλώντας τον constructor της κλάσης School.

.
.
.
.
.
.
.

Στη συνέχεια η rand() αρχικοποιείται δίνοντας ως seed το time(NULL). Η συνάρτηση αυτή θα αποφανθεί αν θα γίνει enter ενός Student ή place ενος Teacher. Η τυχαιότητα πάει ως εξης:

- Αποφασίζεται τυχαία αν θα επεξεργαστούμε Student ή Teacher.
    - Αν επιλεγεί Student, τότε επιλέγεται τυχαία ένας μαθητής. Έπειτα, καλείται η enter με όρισμα τον pointer που δείχνει σε αυτόν. Ο μαθητής αρχίζει και κινείται μέσα στο σχολείο φτάνοντας μέχρι εκεί που μπορεί.
    - Αν επιλεγεί Teacher, τότε επιλέγεται τυχαία ένας καθηγητής. Έπειτα επιλέγεται τυχαία αν θα μπει ή όχι στην τάξη. Αυτό σημαίνει ότι, αν το αποτέλεσμα είναι θετικό τότε καλείται η place με όρισμα τον pointer που δείχνει στον Teacher που επελέγη. Αν το αποτέλεσμα είναι αρνητικό τότε απλά θέτουμε το “flag” member του καθηγητη σε “1” υποδεικνύοντας ότι είναι μέσα στην τάξη χωρίς όμως να είναι. Αυτό ουσιαστικά σημαίνει ότι οι μαθητές θα μπορούν να μπαίνουν στην τάξη τους ακόμα κι αν το flag του καθηγητή είναι “1”, καθώς αυτός δεν βρίσκεται μέσα (αφού δεν κλήθηκε η place).

Εσωτερικά των enter/leave εκτυπώνονται και μηνύματα για τη συνολική διαδρομή των μαθητών.

Στη συνέχεια, εκτυπώνεται ό,τι υπάρχει μέσα στο σχολικό κτίριο, εκτυπώνοντας όλους του επιμέρους χώρους και τέλος γίνονται delete οι πίνακες που περιέχουν τους μαθητές και τους καθηγητές, γεγονός μου ενεργοποιεί τους destructors τους.

ΑΛΛΑ ΔΙΑΔΙΚΑΣΤΙΚΑ & ΣΧΟΛΙΑ

Υπάρχει ένα makefile το οποίο έχει τις εξης εντολές:

- make
    - Δημιουργεί το εκτελέσιμο
- make clean
    - Σβήνει το εκτελέσιμο

Επίσης υπάρχει το studentsFULL.txt και το teachersFULL.txt που είναι απαραίτητα για την εκτέλεση του προγράμματος, καθώς η main είναι βασισμένη σε αυτά (σε περίπτωση που επέλεγα ενα πιο δυναμικό περιβάλλον προφανώς η αντιμετώπιση μου θα ήταν διαφορετική, αλλά υπήρχε ελαστικότητα πάνω σε αυτό.)

*Κατά την εκτέλεση του προγράμματος αρκεί να γράψετε τα εξης:

- make
- ask1 2 3 6 15

(Λόγω του πως έχω δομήσει το πρόγραμμα, είναι απαραίτητο το 1ο όρισμα να είναι ο αριθμός “2” που υποδηλώνει το πόσα άτομα χωράνε σε μια τάξη, καθώς το memory allocation γίνεται με βάση αυτό τον αριθμό και οι μαθητές/καθηγητές είναι τόσοι, ώστε να συμβαδίζουν με αυτό τον αριθμό. Τα υπόλοιπα ορίσματα μπορείτε να τα βάλετε ό,τι θέλετε απλά η συμβουλή είναι για πιο ρεαλιστικά αποτελέσματα. Κάθε φορά που τρέχετε το πρόγραμμα θα πρέπει να δίνει διαφορετικά αποτελέσματα.)

**Η υλοποίηση που ακολούθησα είναι οι μαθητές να μπαίνουν ένας-ένας και να κινούνται μέσα στο σχολείο μέχρι εκεί που μπορούν.