600.315/415

object oriented class/method definition in O2

```
type Phone: tuple (    area_code: integer,
                       number: integer );

type Date: tuple (     year: integer,
                       month: integer,
                       day: integer );

class Person
    type tuple (           ssn: string,
                       name: tuple (       firstname: string,
                                           middlename: string,
                                           lastname: string ),
                                           number: integer,
                       address: tuple (    street: string,
                                           apt_no: string,
                                           city: string,
                                           state: string,
                                           zipcode: string ),
                       birthdate: Date,
                       sex: character )

    method             age: integer
end

class Student Inherit Person
    type tuple (       class: string,
                       majors_in: Department,
                       minors_in: Department,
                       registered_in: set (Section),
                       transcript: set (tuple (     grade: character,
                                                    ngrade: real,
                                                    section: Section ) ) )

    method             grade_point_average: real,
                       change_class: boolean,
                       change_major (new_major: Department): boolean
end

class Grad_Student Inherit Student
    type tuple (       degrees: set (tuple (  college: string,
                                              degree: string,
                                              year: integer ) ),
                       advisor: Faculty )
end

class Faculty Inherit Person
    type tuple (       salary: real,
                       rank: string,
                       foffice: string,
                       fphone: Phone,
                       belongs_to: set (Department),
                       grants: set (Grant),
                       advises: set (Student) )

    method             promote_faculty,
                       give_raise (percent: real)
end
```

```
class Department
    type tuple (       dname: string,
                       office: string,
                       dphone: Phone,
                       members: set (Faculty),
                       majors: set (Student),
                       chairperson: Faculty,
                       courses: set (Course) )

    method             add_major (s: Student),
                       remove_major (s: Student): boolean
end

class Section
    type tuple (       sec_num: integer,
                       qtr: Quarter,
                       year: Year,
                       students: set ( tuple ( stud: Student,
                                               grade: character ) ),
                       course: Course,
                       teacher: Instructor )

    method             change_grade (s: Student, g: character)
end

class Course
    type tuple (       cname: string,
                       cnumber: string,
                       cdescription: string,
                       sections: set (Section),
                       offering_dept: Department )

    method             update_description (new_d: string)
end
```

O2 SQL

```
Q1:select tuple ( fname: s.name.firstname,
                  lname: s.name.lastname )
   from s in Student
   where s.majors_in.dname = "Computer Science"

Q2:select tuple ( fname: s.name.firstname,
                  lname: s.name.lastname )
                  transcript: select tuple (
                              cname: sc.section.course.cname,
                              sec_no: sc.section.sec_num,
                              quarter: sc.section.qtr,
                              year: sc.section.year,
                              grade: sc.grade )
                         from sc in sec )
   from s in Student , sec in s.transcript
   where s.majors_in.dname = "Computer Science"
```

C++
Class reprerentation for Person.

```
class Person: o2_root {
public:
    char*  ssn ;
    struct  {
        char* firstname ;
        char* middlename ;
        char* lastname } name ;
    struct  {
        int    number
        char* street ;
        char* apt_no ;
        char* city ;
        char* state ;
        char* zipcode } address ;
    struct  {
        int  year
        int  month
        int  day } birthdate ;
    char sex ;

    int age () ;
    }
```

```
(a)
method body age: integer in class Person
{    int a ;
     Date d ;
     d = today() ;
     a = d->year - self->birthdate->year ;
     if ( d->month < self->birthdate->month ) ||
        ( ( d->month == self->birthdate->month ) && ( d->day < self->birthdate->day ) )
        --a ;   /* decrements a by 1 */
     return a;
}

method body grade_point_average: real in class Student
{    float sum = 0.0 ;
     int count = 0 ;
     struct {
         char gr ;
         float ngrade ;
         o2_Section sec ;
     } t ;
     for ( t in self->transcript ) {
     sum += t->ngrade ;  ++count ; /* increments sum by ngrade, count by 1 */
         }
         return sum/count ;
}

method body change_major (new_major: Department): boolean in class Student
{    if (self->major-> remove_major (self) ) {
         return 0 ;
         }
     else {
         new_major-> add_major (self) ;
         self->majors_in = new_major ;
         return 1;
         }
}

method body remove_major (s: Student): boolean in class Department
{    if (s in self->majors) {
         self->majors -= set(s) ; /* -= applies set difference to remove object s from set of majors */
         return 1 ;
         }
     else return 0 ;
}

method body add_major (s: Student) in class Department
{    self->majors += set(s) ; /* += applies set union to add object s to set of majors */
}

(b)
name All_Persons: set (Person) /* a persistent root to hold all persistent Person objects */

name John_Smith: Person ; /* a persistent root to hold a single Person object */

run body {
     o2 Person p = new Person ; /* creates a new Person object p */

     *p = tuple (ssn: "333445555",
                 name: tuple (firstname: "Franklin", middlename: "T", lastname: "Wong" ),
                 address: tuple (number: 638, street: "Voss Road", city: "Houston",
                                 state: "Texas", zipcode: "77079" ),
                 birthdate: tuple (year: 1945, month: 12, day: 8 ),
                 sex: M );
     All_Persons += set (p) ; /* p becomes persistent by attaching to persistent root */

        /* now put values in persistent named object John_Smith */
     John_Smith->ssn = "123456789",
     John_Smith->name: tuple (firstname: "John", middlename: "B", lastname: "Smith" ),
     John_Smith->address: tuple (number:731, street: "Fondren Road", city: "Houston",
                                 state: "Texas", zipcode: "77036" ),
     John_Smith->birthdate: tuple (year: 1955, month: 1, day: 9 ),
     John_Smith->sex: M ;
}
```