Homework 4

Name: Brandon Tam
SID: 913892839
Partner Name: Shivang Soni
Partner SID: 915623718

June 5, 2018

Due 11:59PM June 5, 2018. **READ ALL DIRECTIONS VERY CAREFULLY!** Submit your code, tex files along with a generated PDF. **DO NOT SUBMIT DATA FILES!** For this homework you will be working in groups of two, a group of three will only be allowed with approval due to odd number of students. All programs will be evaluated on the CSIF. Upload your files as a tar gzip file (tgz). Only submit one homework per partner. This specification is subject to change.

You are designing a database for a university called FakeU. As a trial you have been provided grade data from courses for departments ABC and DEF. The grade data is from Summer of 1989 until Summer of 2012. The data provided is in CSV format, and is only as complete as could be made possible. There may be errors, omissions or redundant data in the files. FakeU like UC Davis is on a quarter system, however they have recently transitioned to a single summer quarter instead of two summer sessions. This has corrupted some of their summer data as all summer session classes have now been grouped into a single summer quarter term. Each course has a course ID (CID), a term it was offered (TERM), a subject (SUBJ), a course number (CRSE), a section (SEC), and number of units (UNITS). Within a course there listings of meetings, the instructor of the meeting (INSTRUCTOR(S)), meeting type (TYPE), day of meeting (DAYS), time of meeting (TIME), meeting building (BUILD), and meeting room (ROOM) are also listed. For each student that takes the course there is a student seat (SEAT), a student ID (SID), the students surname (SURNAME), the students preferred name (PREFNAME), the students (LEVEL), the number of units the student is receiving (UNITS), the students class standing (CLASS), the students major (MAJOR), the grade the student received in the course (GRADE), the students registration status (STATUS), and the students e-mail address (EMAIL). There may be courses that are cross listed between the two departments (e.g. ABC 123 may be cross listed as DEF 456).

You **MUST** put each problem on a separate page with 1a on the second page, for example 1a will be on page 2 and 1b will be on page 3 (this template is already setup for this). You **MUST** put your name and student ID in the provided author section above. **FAILURE TO DO SO MAY RESULT IN NO CREDIT!** The data will be provided on Canvas, and the CSV files will also be on the CSIF in /home/cjnitta/ecs165a/Grades. All submissions will be compared with MOSS, including against past submissions.

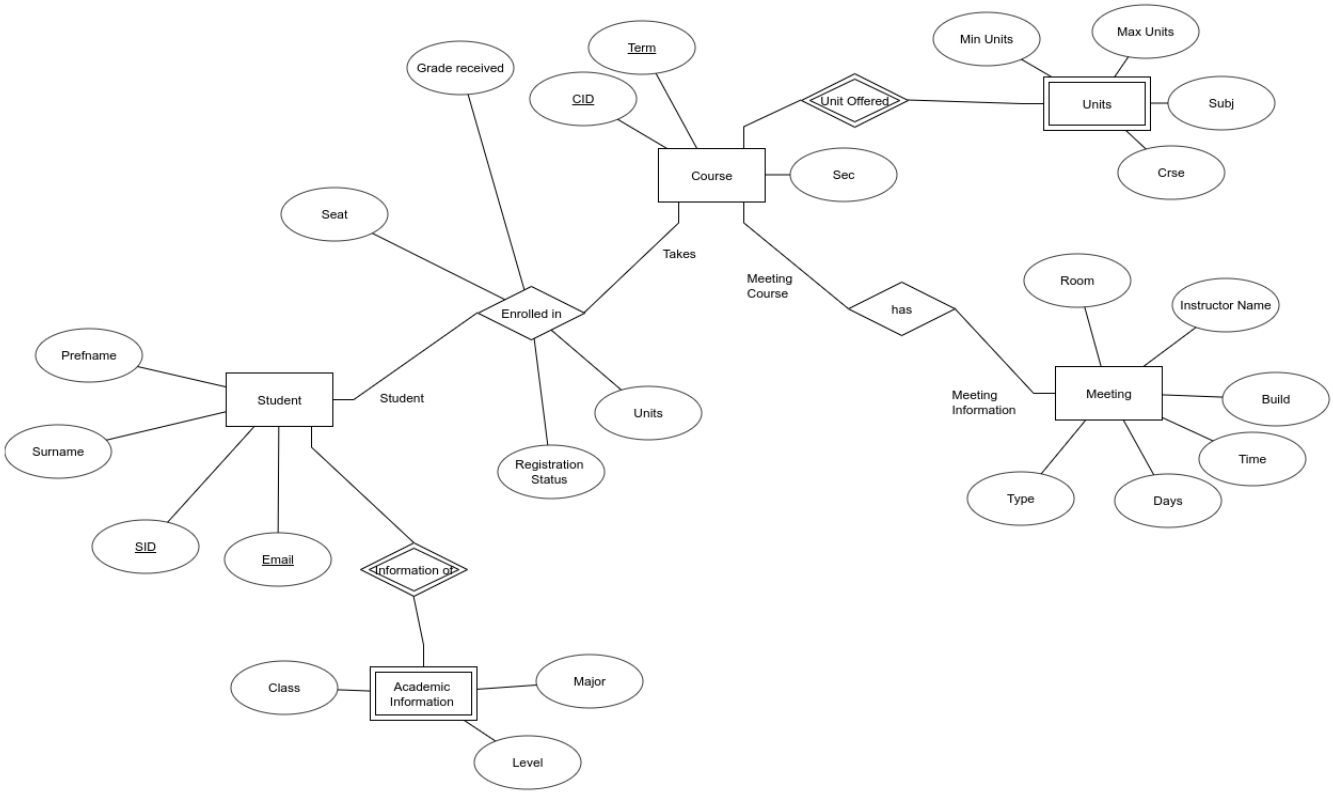
Some useful tips:

- When loading the tuples into the database, insert them in batches. Inserting one tuple at a time may cause the program to take on the order of tens of minutes or hours instead of a few minutes.
- Test a subset of the data first.

Part 1

You will be creating a database schema for your grade data.

a. Provide an ER diagram for your database schema. Only include images generated from vector based programs.



b. Provide a description of the tables in your schema, and their attributes. Make sure you describe how you will store the instructor, student, building, course, etc. information.
Relations for ER Diagram:
$Student(\underline{SID}, Surname, Prefname, \underline{Email})$
$AcademicInfo(Level, \underline{Email}, Major, \underline{SID}, Class)$
$CourseStudent(Seat, Grade, Status, \underline{SID}, \underline{CID}, \underline{Email}, \underline{Term}, Units)$
$Course(\underline{CID}, \underline{Term}, Sec)$
$CourseAdministrative(MinUnits, MaxUnits, Subj, Crse, \underline{CID}, \underline{Term})$
$Meeting(CID, Term, Instructor, Type, Days, Time, Build, Room)$

For the relation Student:
This relation has 4 attributes: SID stored as type Integer, Surname stored as type varchar, Prefname stored as type varchar, and Email stored as type varchar where SID and Email are primary keys.

For the relation AcademicInfo(Stores level,class and major of student):
This relation has 5 attributes: Level stored as type varchar, SID stored as type Integer, Class stored as type varchar, Major stored as type varchar, and Email stored as type varchar where SID and Email are primary keys.

For the relation CourseStudent(This determines relation between two entities student and course):
This relation has 8 attributes: SID stored as type Integer, Email stored as type varchar, Seat stored as type Integer, CID stored as type varchar,Grade stored as type varchar,Status stored as type varchar,Term stored as varchar. and Units stored as type float.

For the relation Course:
This relation has 3 attributes: CID stored as type varchar, Term stored as type integer, and Sec stored as type varchar.

For the relation CourseAdministrative(stores more details of course):
This relation has 6 attributes: CID stored as type varchar, Term stored as type integer, Subj stored as type varchar,MinUnits stored as float,MaxUnits stored as float, and Crse stored as varchar.

For the relation Meeting(Stores detail about the lecture hall,instructor etc):
This relation has 8 attributes: CID stored as type varchar, Term stored as type integer, Instructor stored as type varchar,Type stored as varchar, Days stored as varchar ,Build stored as varchar,Time stored as varchar, and Room stored as varchar.

c. What are the functional (and multivalue) dependencies that you expect to hold for each relation if any. If you don't expect any to hold, describe why not.
$Student(\underline{SID}, Surname, Prefname, \underline{Email})$

Non-trivial functional dependencies expected to hold are

$SID \rightarrow Surname, Prefname, Email$
$Email \rightarrow Surname, Prefname, SID$

Non-trivial multivalue dependencies expected to hold are

$Surname \twoheadrightarrow Prefname$
$Prefname \twoheadrightarrow Surname$

$AcademicInfo(Level, \underline{Email}, Major, \underline{SID}, Class)$

For relation AcademicInfo, non-trivial functional dependencies expected to hold are

$Class \rightarrow Level$
$SID \rightarrow Email, Surname, Prefname$
$Email \rightarrow SID, Surname, Prefname$

Non-trivial multivalue dependencies expected to hold are

$Class \twoheadrightarrow Level$
$Class \twoheadrightarrow Email, Major, SID$
$SID \twoheadrightarrow Email, Surname, Prefname$
$Email \twoheadrightarrow SID, Surname, Prefname$

$CourseStudent(Seat, Grade, Status, \underline{SID}, \underline{CID}, \underline{Email}, \underline{Term}, Units)$

Non-trivial functional dependencies expected to hold in relation CourseStudent are

$Grade \rightarrow Status$
$CID \rightarrow Units$
$SID \rightarrow Email$
$Email \rightarrow SID$
$SID, CID, Email, Term \rightarrow Seat, Grade, Status, Units$

The multivalue dependencies expected to hold are

$Grade \twoheadrightarrow Status$
$Grade \twoheadrightarrow Seat, Grade, SID, CID, Email, Term, Units$
$CID \twoheadrightarrow Units$
$CID \twoheadrightarrow Seat, Grade, Status, SID, Email, Term$
$SID \twoheadrightarrow Email$
$SID \twoheadrightarrow Seat, Grade, Status, CID, Term, Units$
$Email \twoheadrightarrow SID$
$Email \twoheadrightarrow Seat, Grade, Status, CID, Term, Units$
$SID, CID, Email, Term \twoheadrightarrow Seat, Grade, Status, Units$

$Course(\underline{CID}, \underline{Term}, Sec)$

In this relation, only one functional dependency will hold

$CID, Term \rightarrow Sec$

Then the multivalue dependency that should hold is

$CID, Term \twoheadrightarrow Sec$

$CourseAdministrative(MinUnits, MaxUnits, Subj, Crse, \underline{CID}, \underline{Term})$

Functions dependencies expected to hold in relation CourseAdministrative are

$CID \rightarrow MinUnits, MaxUnits, Subj, Crse$
$Crse \rightarrow MinUnits, MaxUnits$
$CID, Term \rightarrow Subj, Crse, MinUnits, MaxUnits$

Multivalue dependencies that should hold then are

$CID \twoheadrightarrow MinUnits$
$CID \twoheadrightarrow MaxUnits$
$CID \twoheadrightarrow Subj$
$CID \twoheadrightarrow Crse$
$CID \twoheadrightarrow MinUnits, MaxUnits$
$CID \twoheadrightarrow MinUnits, Subj$
$CID \twoheadrightarrow MinUnits, Crse$
$CID \twoheadrightarrow MaxUnits, Subj$
$CID \twoheadrightarrow MaxUnits, Crse$
$CID \twoheadrightarrow Subj, Crse$
$CID \twoheadrightarrow MinUnits, MaxUnits, Subj$
$CID \twoheadrightarrow MinUnits, MaxUnits, Crse$
$CID \twoheadrightarrow MaxUnits, Subj, Crse$
$CID \twoheadrightarrow MinUnits, MaxUnits, Subj, Crse$

$CID \twoheadrightarrow MaxUnits, Subj, Crse, Term$
$CID \twoheadrightarrow MinUnits, Subj, Crse, Term$
$CID \twoheadrightarrow MinUnits, MaxUnits, Crse, Term$
$CID \twoheadrightarrow MinUnits, MaxUnits, Subj, Term$
$CID \twoheadrightarrow Subj, Crse, Term$
$CID \twoheadrightarrow MaxUnits, Crse, Term$
$CID \twoheadrightarrow MaxUnits, Subj, Term$
$CID \twoheadrightarrow MinUnits, Crse, Term$
$CID \twoheadrightarrow MinUnits, Subj, Term$
$CID \twoheadrightarrow MinUnits, MaxUnits, Term$
$CID \twoheadrightarrow Crse, Term$
$CID \twoheadrightarrow Subj, Term$
$CID \twoheadrightarrow MinUnits, Term$
$CID \twoheadrightarrow Term$

$Crse \twoheadrightarrow MinUnits$
$Crse \twoheadrightarrow MaxUnits$
$Crse \twoheadrightarrow MinUnits, MaxUnits$

$Crse \twoheadrightarrow MaxUnits, Subj, Term$
$Crse \twoheadrightarrow MinUnits, Subj, Term$
$Crse \twoheadrightarrow Subj, Term$

$CID, Term \twoheadrightarrow Subj, Crse, MinUnits, MaxUnits$ —————————————————————

———

$Meeting(CID, Term, Instructor, Type, Days, Time, Build, Room)$

No functional dependencies or multivalued dependencies are expected to hold for the following relation. This is because there are many possibilities of a repeated course being taught in the same term by the same instructor. Days, time, build, and room have many possibilities which also creates many possibilities. Therefore, the only function dependency expected to hold is all attributes together.

There are many multivalue dependencies that are expected to hold for this relation. The attributes in this relation have so many possible combinations that there are 2^8 multivalue dependencies that could hold. Some example multivalue dependencies expected to hold are

$CID \twoheadrightarrow Term, Instructor, Type, Days, Time, Build, Room$
$Term \twoheadrightarrow CID, Term, Type, Days, Time, Build, Room$
$Instructor \twoheadrightarrow CID, Term, Type, Days, Time, Build, Room$
$Type \twoheadrightarrow CID, Term, Instructor, Days, Time, Build, Room$
$Days \twoheadrightarrow CID, Term, Instructor, Type, Time, Build, Room\, Time \twoheadrightarrow CID, Term, Type, Days, Days, Build, Room$
$Build \twoheadrightarrow CID, Term, Instructor, Type, Days, Time, Room$
$Room \twoheadrightarrow CID, Term, Instructor, Type, Days, Time, Build$
$CID, Term \twoheadrightarrow Instructor$
$CID, Instructor \twoheadrightarrow Term$
$CID, Term, Type, Days, Time, Build, Room \twoheadrightarrow Instructor$
$Instructor, Term, Type, Days, Time, Build, Room \twoheadrightarrow CID$

Part 2

Write a program to load the grade data into a PostgreSQL database called FakeUData that follows your schema. You **MUST** use the database called FakeUData, and should assume it will already be created for you without any tables or data in it. You may **NOT** hardcode usernames in your code, use the USER environmental variable instead if user is needed. Your program can be written in C++ or python, you may **NOT** use standalone SQL or text files that hold your queries. You may **NOT** use shell calls to implement your program. All your queries need to be in your code. If you choose to make a C++ program, you must include a makefile and call the program loadfakeu. Include a readme file with descriptions of any issues/problems. If you choose to make a python program you must specify which version of python you used, and must provide a loadfakeu bash script to launch your python program. The loadfakeu program **MUST** be able to take one optional argument (the directory where the CSV data files will be located). If the argument is omitted, the default is the current working directory. Scripts that require greater than 10 minutes to load all of the data may lose points.

Part 3

Write another program to query your database to calculate the following values, put the results in your write up, some may be best described with a chart instead of raw values. Name your program queryfakeu, it must output the data values for the following queries. The query program does not have to do everything in the SQL queries, but should limit the amount of data transfered. For example it is acceptable to have one SQL query for each unit number (1 - 20) for 3a, but it would be unacceptable to pull all student data on a per student basis and calculate the results.

a. Calculate the percent of students that attempt 1 20 units of ABC or DEF per quarter for every unit increment (e.g. 1, 2, 3,).

Units	Percentage of Students That Attempted Units
1	1.16%
2	1.5%
3	2.42%
4	10.54%
5	2.12%
6	1.31%
7	0.63%
8	2.25%
9	1.08%
10	0.59%
11	0.71%
12	3.65%
13	1.09%
14	0.28%
15	0.25%
16	0.47%
17	0.11%
18	0.34%
19	0.07%
20	0.4%

b. Find the easiest and hardest instructors based upon the grades of all the students they have taught in their courses. Provide their name and the average grade they assigned. (Ignore P/NP, S/NS grades)

Instructor marking scheme	Instructor Name	AVG GPA
Easy	Murphy, Melanie S.	4
Easy	Powell, Liliana M.	4
Easy	White, Sophia V.	4
Easy	Porter, Ryan E.	4
Easy	Thomas, Santiago G.	4
Easy	Mendoza, Anthony A.	4
Easy	Odonnell, Madison G.	4
Easy	Smith, Daniel N.	4
Easy	Bell, David Q.	4
Easy	Calderon, Isabella A.	4
Easy	Norris, Nathan A.	4
Easy	Henderson, Alexa M.	4
Easy	Brooks, Michael Y.	4
Easy	Mckay, Isaac A.	4
Easy	Gordon, Noah R.	4
Easy	Hart, Vincent B.	4
Easy	Mcclure, Noah P.	4
Easy	Hayes, Aaliyah D.	4
Easy	Allen, Mateo G.	4
Easy	Taylor, Liam J.	4
Easy	Turner, Emily A.	4
Easy	Fletcher, Arianna J.	4
Easy	Russo, Angel J.	4
Easy	Morris, Ariana O.	4
Easy	Walker, Evelyn K.	4
Easy	Jackson, Dominic E.	4
Difficult	Houston, Noah J.	0
Difficult	Morris, Wyatt J.	0
Difficult	Williamson, Jasmine B.	0
Difficult	Love, Jeremiah A.	0

c. Calculate the average GPA for the students that take each number of units from part a. Assume that the grades have standard grade points (A+ = 4.0, A = 4.0, A- = 3.7, B+ = 3.3).

Units	Average GPA of Students That Attempted Units
1	3.67
2	3.58
3	3.21
4	3.12
5	3.23
6	3.63
7	3.73
8	3.26
9	3.47
10	3.19
11	4.0
12	3.46
13	3.0
14	3.75
15	3.44
16	3.57
17	3.98
18	3.80
19	3.0
20	3.57

d. Find the courses with the highest and lowest pass rates. Assume that F, NP, and NS are not passing grades.
Courses with highest passing rate:

ABC364 DEF359 ABC250 DEF423 ABC349 DEF241 DEF363 DEF404 DEF329 ABC325 ABC246 ABC372 ABC353 DEF263 DEF247 DEF365 ABC368 DEF381 ABC333 DEF322 ABC206 DEF272 DEF360 DEF210 DEF233 DEF358 ABC322 DEF284 ABC243 DEF338 DEF380 DEF321 ABC302 DEF266 DEF400 ABC311 ABC228 ABC304 DEF410 DEF309 ABC366 DEF399 ABC309 DEF394 DEF343 DEF361 DEF260 DEF302 DEF346 DEF282 DEF334 DEF418 DEF419 ABC323 ABC359 ABC313 ABC233 ABC365 DEF420 ABC310 DEF328 DEF264 DEF372 DEF265 ABC341 DEF421 ABC318 DEF339 DEF283 ABC208 ABC231 ABC247 ABC316 DEF296 DEF295 ABC324 ABC377 DEF267 DEF350 ABC301 ABC336 ABC225 DEF274 DEF268 DEF387 ABC362 ABC212 DEF362 DEF407 DEF224 ABC203 DEF262 DEF390 DEF231 DEF401 ABC348 ABC314 ABC335 ABC219 ABC369 ABC352 ABC235 ABC356 DEF287 ABC102 DEF271 ABC205 DEF345 DEF246 DEF383 DEF333 ABC340 DEF376 DEF388 ABC237 DEF385 DEF353 DEF316 DEF374 DEF319 DEF212 DEF317 DEF405 DEF351 DEF375 ABC258 DEF217 DEF280 DEF223 ABC331 DEF281 ABC338 DEF311 DEF297 ABC343 ABC350 DEF327 ABC307 DEF232 DEF398 ABC346 DEF213 ABC373 ABC320 DEF395 DEF379 DEF270 DEF228 ABC332 ABC249 DEF275 DEF289 ABC240 DEF303 DEF352 ABC259 DEF337 DEF222 DEF425 DEF424 DEF340 DEF291 ABC306 DEF306 ABC317 ABC260 ABC354 DEF326 ABC224 ABC315 ABC367 DEF313 DEF240 DEF202 DEF344 ABC110 DEF335 DEF314 ABC360 DEF105 ABC339 DEF368 DEF218 DEF397 DEF102 ABC337 DEF277 DEF273 DEF402 DEF354 ABC312 ABC363 DEF227 DEF382 ABC326 ABC351 DEF301 ABC355 ABC344 DEF288 DEF377 DEF290 DEF237 DEF349 DEF366 ABC305 ABC328 DEF325 ABC345 ABC347 ABC255 ABC204 DEF336 DEF225 DEF226 ABC236 ABC303 ABC319 DEF371 DEF236 ABC215 DEF348 DEF323 DEF278 DEF315 ABC308 DEF261 DEF408 ABC207 ABC361 ABC334 ABC223 DEF312 DEF364 DEF370 ABC374 DEF310 DEF384 DEF276 DEF392 DEF427 DEF305 ABC242 ABC342 DEF378 DEF320 DEF386 DEF357 DEF298 DEF244 DEF396 ABC357 DEF269 ABC109 DEF324 DEF216 ABC327 DEF355 DEF331 DEF342 DEF403 ABC254 DEF422 ABC218

Courses with lowest passing rate:

DEF307 DEF207 ABC214 DEF341

e. Find the list of courses that must be cross listed as they have the same meeting times during the normal quarters. Only list the pair once, put the course name/number string in alphabetically order of the pairs.

LIST OF COURSES THAT ARE CROSSLISTED
106
110
112
113
225
245
251
253
254
256
261
285
286
287
300
371
376
378
379
416
426

f. Find the major that performs the best/worst on average in ABC courses. Repeat the analysis for DEF courses as well.
ABC2, O239, OT31, OT99, OT26, O223, OTH1, OT30, O251, O194 ,OT39 ,OTH6, OTH8, O238, OTH3 ,OTH9, O257, OTH4, OT48, O155, O218, OT71, OT54, OT82, O143, O130, O192, O225, OT40, O224, OT14, OT93, O174, O110, O189, OT11, O187, ABC1, OT67, O241 ,OT94, OT83, O248, O198, ,O214, OT55, O118, OT17, O108, OT28, O227, OT22, OT59, O112, O127, O200, OT73, O123 ,OT77, O183, DEF2, O240, O190, DEF1, O278, OT16, OT44, O252, OT69, OTH2, ABCG, O101, O244, OT21, OT80, O135, O216, OT72, O160, O188, OT42, O284, O262, OT68, OTH5, OT76, O107, OT20, OT60, O146, O212, O164, OT98, OT25, OT27, OT34, OTH7, O181, O159, O115, OT75, OT35, OT12, O204, OT37, O103, OT64, O161, O268, O131, O250, O253, O116, O104, DEFG, O184, OT15, OT62, O243, OT13, O237, OT74, O254, OT10, OT78, OT56, OT41, O117

g. Find the top 5 majors that students transfer from into ABC. What is the percent of students from each of those majors compared to overall transfers?
.

Major	Percentage of Students transfer from to ABC
DEF2	74.5629140909743%
OT35	8.00156442662663%
DEF1	5.59421956659578%
DEFG	4.9412700778369%
OTH7	2.26566613081383%

h. Find the top 5 majors that students transfer to from ABC. What is the percent of students to each of those majors compared to overall transfers out?
.

Major	Percentage of Students transfer from ABC to other major
DEF2	74.5629140909743%
OT35	8.00156442662663%
DEF1	5.59421956659578%
DEFG	4.9412700778369%
OTH7	2.26566613081383%

Part 4

Extra credit: The Efficient XML Interchange (EXI) is a format for the compact representation of XML information. The CSV files provided for this assignment have been consolidated into a single EXI file (HW4Grades.exi) that is available in the resources section of Canvas. Implement a separate program that it can load the database from the EXI file. You may **NOT** use shell calls, or creation of external temporary files for this part. Name your program or bash script loadfakeuexi.

Part 5

Extra credit: Additional queries/query program.

a. Find the courses that appear to be prerequisites for ABC 203, ABC 210, and ABC 222. For this problem list the courses that the X% of students have taken for every 5% increment from 50% - 100% prior to taking the course. (Add this output to your query program.)

b. Write a program that will find an open room for course expansion. The program must prompt for term, CID, and number students to add. The room(s) returned should be ordered from best to worst fit with up to 5 results. Assume that each room capacity is the maximum number of students listed for any particular meeting in the data files (don't forget that lectures may be split across multiple CIDs). Name this program findroomfakeu.