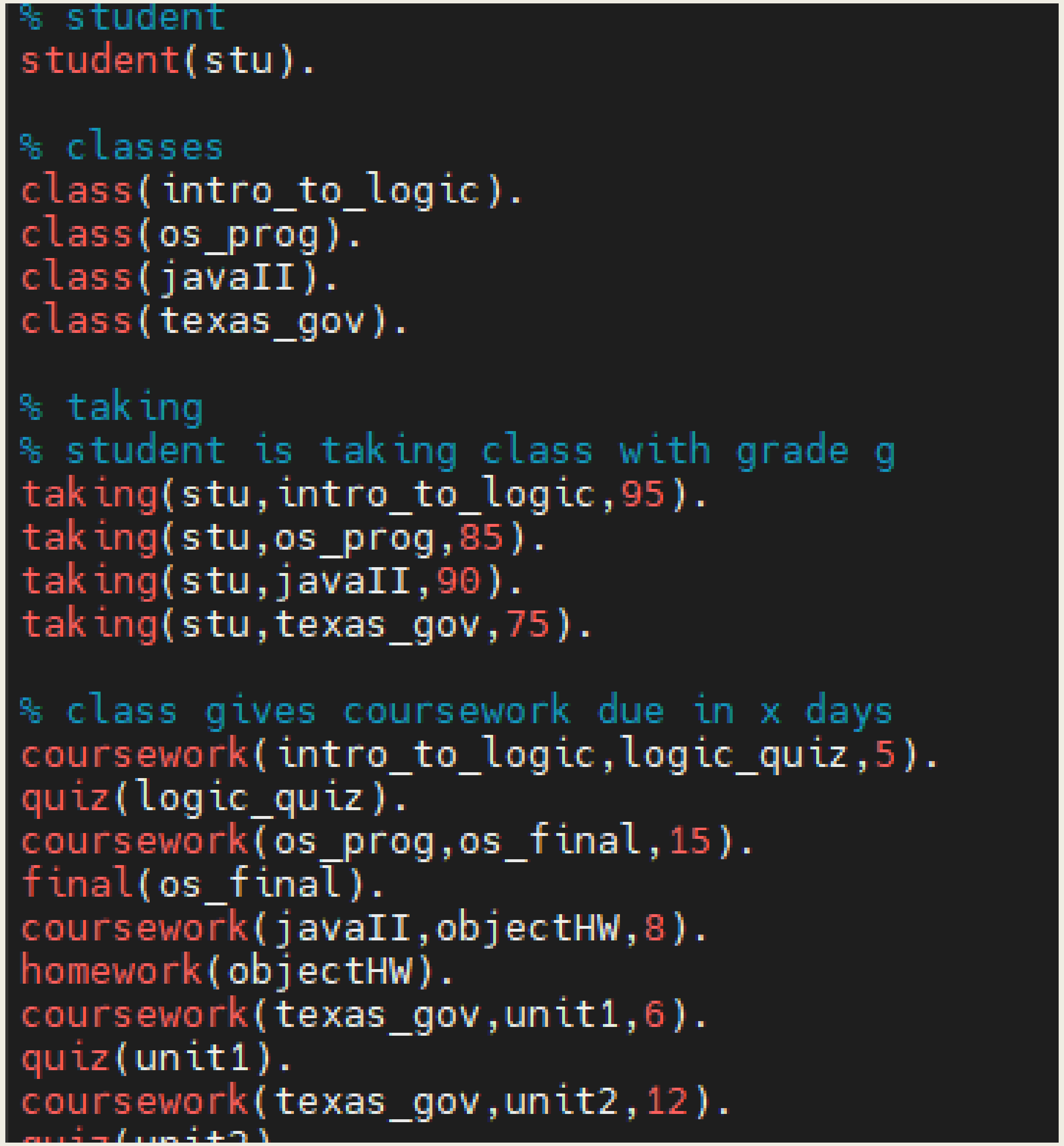
STUDY HELPER

Aru Gyani, James Hooper, Ivan Payne, Richard Gatchalian



```
% student
student(stu).

% classes
class(intro_to_logic).
class(os_prog).
class(javaII).
class(texas_gov).

% taking
% student is taking class with grade g
taking(stu,intro_to_logic,95).
taking(stu,os_prog,85).
taking(stu,javaII,90).
taking(stu,texas_gov,75).

% class gives coursework due in x days
coursework(intro_to_logic,logic_quiz,5).
quiz(logic_quiz).
coursework(os_prog,os_final,15).
final(os_final).
coursework(javaII,objectHW,8).
homework(objectHW).
coursework(texas_gov,unit1,6).
quiz(unit1).
coursework(texas_gov,unit2,12).
```

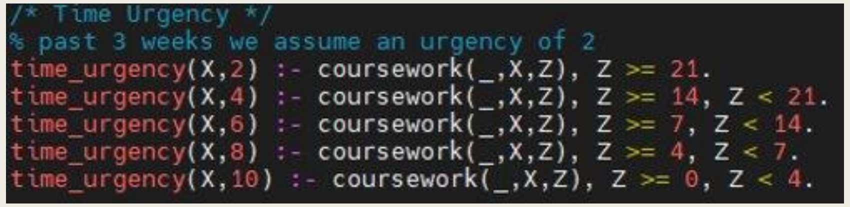
Predicate

Organizing the information from the student was crucial. Creating predicates to organize and store the class information allows for more reasonable and efficient decision making.

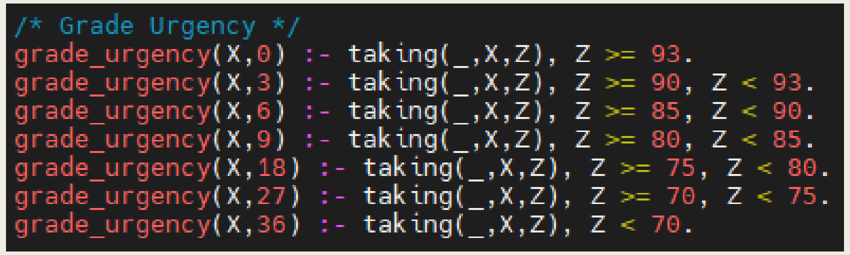
Urgency Rating

In order to create a program that ranks and sorts assignments to complete, we needed a system to rate each assignment.

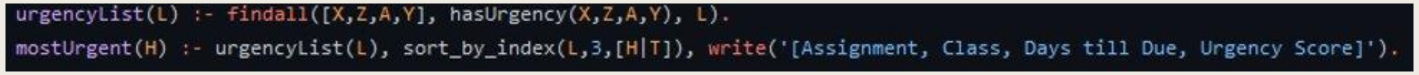
Urgency, is a rating system that takes in factors such as due date, class grade, assignment type, and exam type.



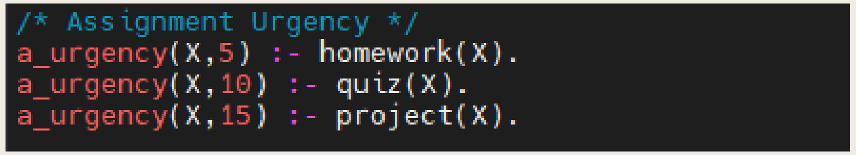
```
/* Time Urgency */
% past 3 weeks we assume an urgency of 2
time_urgency(X,2) :- coursework(_,X,Z), Z >= 21.
time_urgency(X,4) :- coursework(_,X,Z), Z >= 14, Z < 21.
time_urgency(X,6) :- coursework(_,X,Z), Z >= 7, Z < 14.
time_urgency(X,8) :- coursework(_,X,Z), Z >= 4, Z < 7.
time_urgency(X,10) :- coursework(_,X,Z), Z >= 0, Z < 4.
```



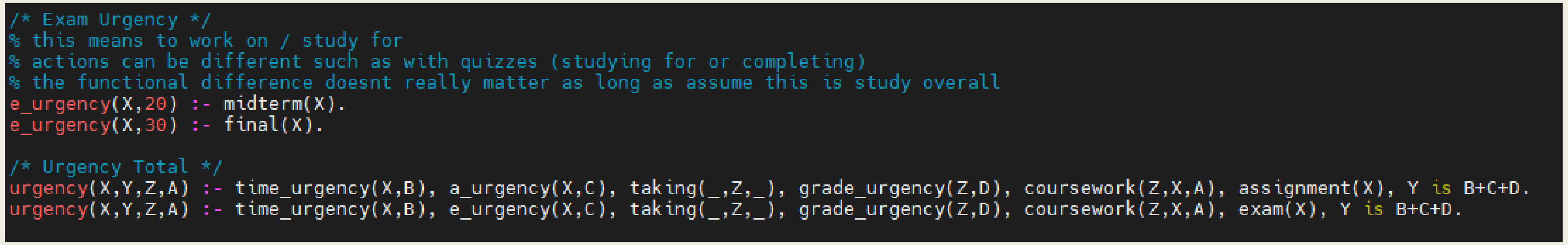
```
/* Grade Urgency */
grade_urgency(X,0) :- taking(_,X,Z), Z >= 93.
grade_urgency(X,3) :- taking(_,X,Z), Z >= 90, Z < 93.
grade_urgency(X,6) :- taking(_,X,Z), Z >= 85, Z < 90.
grade_urgency(X,9) :- taking(_,X,Z), Z >= 80, Z < 85.
grade_urgency(X,18) :- taking(_,X,Z), Z >= 75, Z < 80.
grade_urgency(X,27) :- taking(_,X,Z), Z >= 70, Z < 75.
grade_urgency(X,36) :- taking(_,X,Z), Z < 70.
```



```
urgencyList(L) :- findall([X,Z,A,Y], hasUrgency(X,Z,A,Y), L).
mostUrgent(H) :- urgencyList(L), sort_by_index(L,3,[H|T]), write('[Assignment, Class, Days till Due, Urgency Score]').
```



```
/* Assignment Urgency */
a_urgency(X,5) :- homework(X).
a_urgency(X,10) :- quiz(X).
a_urgency(X,15) :- project(X).
```



```
/* Exam Urgency */
% this means to work on / study for
% actions can be different such as with quizzes (studying for or completing)
% the functional difference doesnt really matter as long as assume this is study overall
e_urgency(X,20) :- midterm(X).
e_urgency(X,30) :- final(X).

/* Urgency Total */
urgency(X,Y,Z,A) :- time_urgency(X,B), a_urgency(X,C), taking(_,Z,_), grade_urgency(Z,D), coursework(Z,X,A), assignment(X), Y is B+C+D.
urgency(X,Y,Z,A) :- time_urgency(X,B), e_urgency(X,C), taking(_,Z,_), grade_urgency(Z,D), coursework(Z,X,A), exam(X), Y is B+C+D.
```

Future improvements.

01

Incorporate algorithm to parse urgency lists and create daily, hourly based time schedules.

02

Allow for multiple student support. Given the time constraints, support for only one student per file is included, however in an ideal situation this will be scalable.

03

Include greater variation in scheduling such as daily hour limits, study breaks, etc.