IT-205 : CAPSTONE PROJECT

SYNTAX SAMURAI

202301233 - Nakum Aayush
202301444 - Vagh Divyesh
202304001 - Aasha Chanpa
202304004 - Hiral Parmar

Pseudocode:

- A class named “subject” is created.
- The public attributes of this class are : code, name, credit, avb (available), prof (professor), venue , core, on.
- Elective (vector of subjects)
- Constructors initialize the default values to it’s attributes.
- A 3D array named "tableatumn" is created to store subjects.
- A 2D vector named "sub" is made to store subjects for each semester.

- function weeklecvec(n):
totall = 0
for each subject in sub[n]:
totall += subject.credit
// To count the total credits for a subject
- A vector named "weeklec" is declared to store subjects for the week.
weeklec.resize(totall)
// To resize the vector to the updated total credit
k = 0
// The iterator k is initialized to 0
for i from 0 to totall-1:
if sub[n][k].avb > 0:
// To iterate over the subject[n] and check if available credits for subject k is greater than 0 so that the lecture can be assigned.
add sub[n][k] to weeklec
// To add the subject k to the weeklec vector.
sub[n][k].avb = sub[n][k].avb - 1
// To decrease the available credits for subject k by 1 , which signifies that the lec has been assigned.
end if
k = sub[n].size() - 1:
k=0
// To check if k has reached the end of sub[n] or not so that it can be set to 0 again
else:
k = k + 1
end if
end for

return weeklec

- TIME COMPLEXITY :- O(totall) , as it will traverse through totall.

- Function isprofavailable(faculty, i, j, k):

if i + 1 < 5 AND faculty equals tableatumn[k][i + 1][j].prof:
return false
// Professor is not available because he has a lecture before/after this slot

else if i > 0 AND faculty equals tableatumn[k][i - 1][j].prof:
return false
// It checks all the slots , if anyone matches , it returns false as professor is not available
else:
for each row in tableatumn:
if faculty equals tableatumn[i][j][k].prof:
return false
// Checks all the rows to find out if professor is available or not in the current time slot
end for
return true
// Otherwise the professor is available , so it will return true

- TIME COMPLEXITY :-O(rows).

- function isvenueavailable(venue, i, j, k):
for each row in tableatumn:
if venue equals tableatumn[i][j][k].venue:
return false
// To check if Venue is available or not in the current time slot
end for

return true
// Venue is available

- TIME COMPLEXITY :- O(rows).

- function islecreq(int i, int j, int k):
max = 0
for each slot in tableatumn[i][j][k]:
max += slot.on
//To calculate total number of classes on a given day

if max > 4:
return false

// Number of classes in that particular time slot exceeds the maximum limit
else:
return true
//Means that the slot is availaOble.

- TIME COMPLEXITY :-O(n)

- function iselelecavailable(faculty, venue,i,j,k):
for each elective_subject in tableatumn[i][j][k].elective:
if faculty equals elective_subject.prof OR venue equals elective_subject.venue:
return false
// Elective subject will not be available due to the same professor or venue in the regular timetable

return true
// If above is false then slot for elective subject is available

- function freeslotassign(i):
for each semester from i to the end of sub:
// Get the subjects for the week for the current semester
weeklec = weeklecvec(i)

for each subject in weeklec:
for each day (j) from 0 to 4:
for each hour (k) from 0 to 4:
br = 0
// Check if the current slot is suitable for the subject
if isprofavailable(subject.prof, i, j, k) and isvanueavailable(subject.vanue, i, j, k) and islecreq(i, j, k) and not subject.core:
// Assign the subject to the current slot in the timetable
tableatumn[i][j][k] = subject
// Set the flag to indicate a slot has been assigned
br = 1
break
else if islectogather(subject.code, subject.prof, subject.vanue, i, j, k): // Assign the subject to the current slot in the timetable
tableatumn[i][j][k] = subject
// Set the flag to indicate a slot has been assigned
br = 1
break
else if subject.core and iselelecavailable(subject.prof, subject.vanue, i, j, k): // Assign the subject as an elective to the current slot in the timetable
tableatumn[i][j][k].elective.push_back(subject)

```
                // Set the flag to indicate a slot has been assigned
                    br = 1
                    break
            else if subject.core and tableatumn[i][j][k].elective.size() > 0:
            // Check if the core subject matches any existing elective subjects in the slot
                    for each elective_subject in tableatumn[i][j][k].elective:
                      if elective_subject.prof == subject.prof and
                        elective_subject.vanue == subject.vanue and
                        elective_subject.code == subject.code:
            // Assign the core subject as an elective to the current slot in the timetable
                        tableatumn[i][j][k].elective.push_back(subject)
                         // Set the flag to indicate a slot has been assigned
                        br = 1
                        break
                  // Break the loop if a slot has been assigned
                  if br:
                     break
                // Break the loop if a slot has been assigned
                if br:
                   break
              // Break the loop if a slot has been assigned
              if br:
                 break
```

- TIME COMPLEXITY :-

  Simplifying: O(i* j * k)

- function islectogather(code, faculty, venue, i, j, k):

```
    for each semester in subject
// Check if subject with the same code, faculty,and venue is scheduled in the current slot
        if tableatumn[i][j][k] exists and
          code == tableatumn[i][j][k].code and
          faculty == tableatumn[i][j][k].prof and
          venue == tableatumn[i][j][k].vanue:
           return true
     // Matching subject found, subject can be assigned to the slot
        return false
    // No matching subject found, subject cannot be assigned to the slot
```

- TIME COMPLEXITY :-O(N) - Linear Time Complexity
- function displaytableindividual(i):

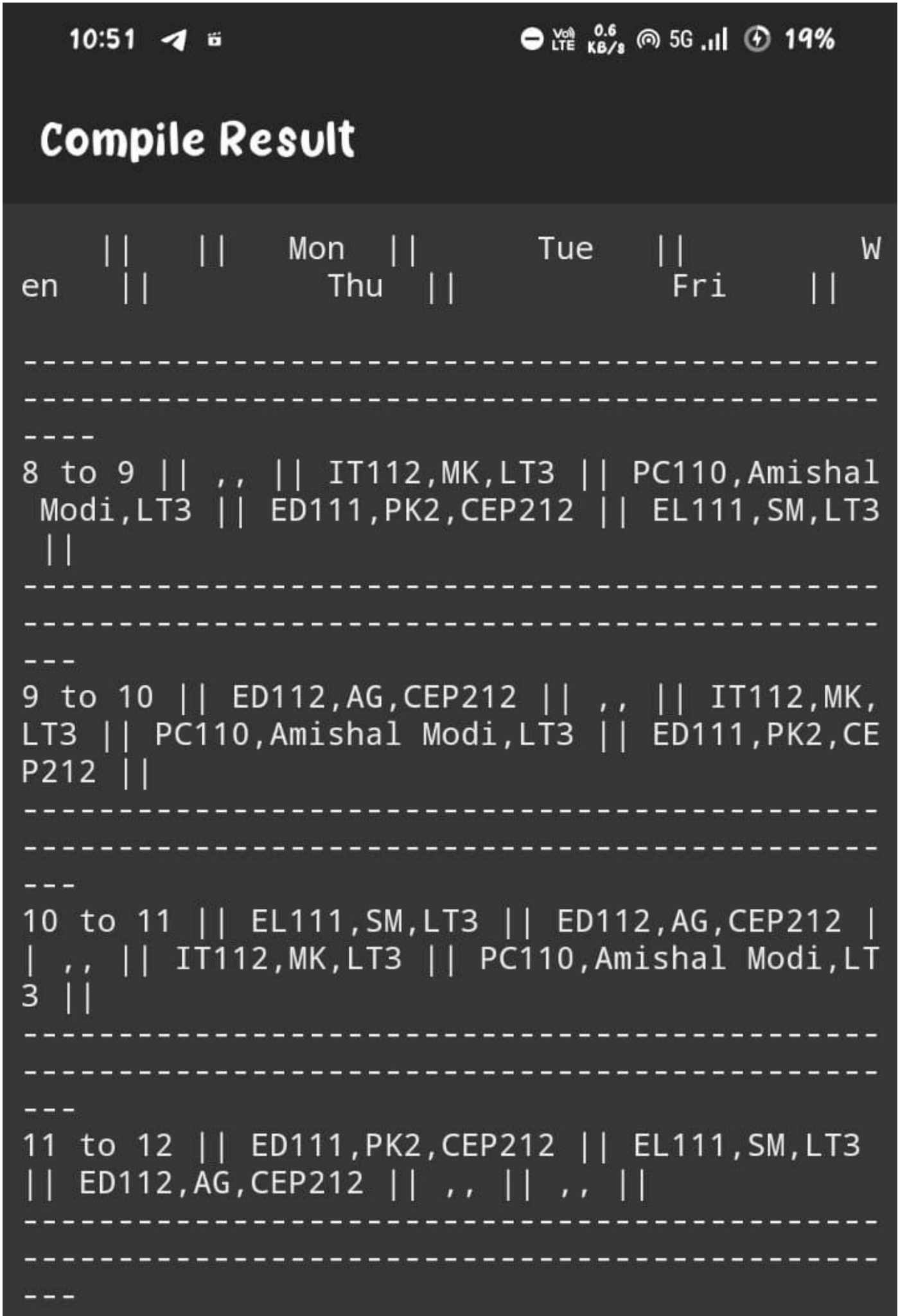
```
    // Print day names
    for each d in day:
       print d + "\t"
    // Iterate through each time slot
    for each time slot (j) in time:
```

```
        // Print time slot label
        print time[j]

        for each hour (k) from 0 to 4:
          // Check if the slot is not empty and not a core subject
            if tableatumn[i][j][k] exists and not tableatumn[i][j][k].core:
          // Print subject details
                print tableatumn[i][j][k].code + "," + tableatumn[i][j][k].prof + "," +
tableatumn[i][j][k].vanue + " || "
          // Check if the slot is a core subject
            else if tableatumn[i][j][k].core:
          // Iterate through each elective subject associated with the core subject
                for each elective subject (s) in tableatumn[i][j][k].elective:
         // Print elective subject details
                    print elective_subject.code + "," + elective_subject.prof + "," +
elective_subject.vanue + " || "
                print "\n" // Add a new line after printing all elective subjects
            // Check if the slot is empty
            else if not tableatumn[i][j][k].on:
                // Print a placeholder
                print " || || "

        print
"\n----------------------------------------------------------------------------------------------------------------------
- ----------------------------------------------"
```

- TIME COMPLEXITY :- O(N * M) - Linear to Quadratic Time Complexity


10:51
0.6 KB/s
5G
19%
Compile Result
|| || Mon || Tue || W
en || Thu || Fri ||
8 to 9 || ,, || IT112,MK,LT3 || PC110,Amishal
Modi,LT3 || ED111,PK2,CEP212 || EL111,SM,LT3
||
9 to 10 || ED112,AG,CEP212 || ,, || IT112,MK,
LT3 || PC110,Amishal Modi,LT3 || ED111,PK2,CE
P212 ||
10 to 11 || EL111,SM,LT3 || ED112,AG,CEP212 |
| ,, || IT112,MK,LT3 || PC110,Amishal Modi,LT
3 ||
11 to 12 || ED111,PK2,CEP212 || EL111,SM,LT3
|| ED112,AG,CEP212 || ,, || ,, ||