```c
/*
Mehmet Celebi
0323916
VLSI Circuit Systems Design
C Programming Project
University of Rome "Tor Vergata"
*/


/*including necessary library
stdio.h for standard of input output library
*/
#include <stdio.h>



/* Structure is data type allows group together different variables
String implementation is failed. Since char used for name.
*/
struct Course {
    char name[100];
    int cfu;
    int grade;
    int withLode;
};

int main() {
    // index value of loop data gathering.
    int numCourses = 0;

    //courses array for Course structure. This array has size of 20
    struct Course courses[20];

    //defining variables will used in main
    int totalGrade = 0; //total obtained Grade*CFU for each course
    int totalCFU = 0; //total obtained CFU (credits)
    int totalLodes = 0; //total LodePoints - method explained below
    int nineLodes = 0; //total number of courses with 9 or 12 CFU
    int sixLodes = 0; //total number of courses with 6 CFU


    /*
    While instruction used for creating loop for gathering informations
    we use condition 1-TRUE to create endless-loop so loop can only trigger
    with user input
    */
    while (1) {

        /*
        endless loop will count numCourses (index of array) each turn and gets input
        from user for name (as char), CFU, Grade and Lode status.

        First loop has exit algorithm to allow user delete current input and
        finish entering input.

        otherwise array will end in 21th input.
        */

        printf("Enter the course name: "); //printing text to terminal
        scanf(" %s", courses[numCourses].name); //scanning inputs of user for name.
                                                //" %s" argument continues scanning until space " "  input

        printf("Enter the CFU: "); //printing text to terminal
        scanf("%d", &courses[numCourses].cfu); //" %d" format specifier for defining integer.

        while (1) {
```

```
67              /* Second while loop to create another loop sequence. Since in Tor Vergata maximum score is 30
68                 and scores lower than 18 is not written to the system. If user tries to enter these values
69                 program will give output and ask user to enter again.
70              */
71              printf("Enter the grade: "); //printing text to terminal
72              scanf("%d", &courses[numCourses].grade); // format specifier for integers
73
74              if (courses[numCourses].grade >= 18 && courses[numCourses].grade <= 30) {
75                  //if grade is in normal range. we break while loop to continue program.
76                  break;
77              }  else if (courses[numCourses].grade < 18) { // lower than 18
78                  printf("Not Passed \\n");
79              }
80              else { // since there is no other option. we use else command to specify other errors.
81                  printf("Invalid grade input. Please enter a grade between 1 and 30.\\n");
82              }
83          }
84
85          if (courses[numCourses].grade == 30) { // e Lode is honour system if student have extraordinary
performance.
86              // e Lode is given only with 30 points so we add this condition to decrease unnecessary input
requests.
87              printf("Enter 1 if with Lode (Honour), 0 otherwise: "); // we used integer since we will have
arithmetic
88              scanf("%d", &courses[numCourses].withLode);             // operation after.
89          }
90
91          numCourses++;
92          /*While loop preferred rather than for loop since counter increase and decrease instructions can be
given in
93          anywhere of the code*/
94
95
96          /*
97          algorithm for deleting current course.
98          Resetting structure variable did not preferred since scanf instruction directly access to memory
99          & notation in scanf retrieves memory address and directly changes data itself than having
100         reference from memory.
101         */
102         char deleteCourse;
103         printf("Would you like to delete the last course added? (y/n): ");
104         scanf(" %c", &deleteCourse); //%c format specifier defines character.
105         if (deleteCourse == 'y' || deleteCourse == 'Y') { //when user presses y or Y in keyboard
106             numCourses--; //index number was increased in line 90 but now it returned current phase
107         }                 //in next iteration same index value will be processed
108
109         /*
110         algorithm for exiting loop.
111         Same approach with deleting algorithm.
112         */
113         char moreCourses;
114         printf("Add another course? (y/n): ");
115         scanf(" %c", &moreCourses); // direct access to memory with &
116         if (moreCourses != 'y' && moreCourses != 'Y') {
117             break; // breaks the main loop
118         }
119     }
120
121     printf("All notes entered: \n");
122
123     for (int i = 0; i < numCourses; i++) { //prints all courses between 0 to last numCourses
124         // since we are no longer changing data of array. Direct access & is no longer needed.
125         // We can process our values with reference from memory
126         printf("Course Name: %s     || ", courses[i].name);
127         printf("CFU: %d     || ", courses[i].cfu);
128         printf("Grade: %d   || ", courses[i].grade);
```

```
129
130         // only prints Lode if grade is 30 to avoid lost data and clearer terminal
131         if (courses[i].grade == 30) {
132             printf("With Lode: %s ", courses[i].withLode ? "e Lode" : " ");
133         }
134
135         //summing CFU and total weighted grade of each course
136         totalCFU = totalCFU + courses[i].cfu;
137         totalGrade = totalGrade + (courses[i].cfu*courses[i].grade);
138
139         printf("\n"); //prints new line
140
141     }
142     //we use float since weighted grade can be decimal.
143
144     // Weighted Grade = Sum of (grade*CFU of each class) / total CFU
145     float weightedGrade = (float)totalGrade / totalCFU; // converts grade to float since it is integer
146     printf("Weighted Grade: %.4f \n", weightedGrade); //%.2f prints float as .XX format
147
148     /*
149     Tor Vergata Mechatronics Engineering Department gives 2 extra point for graduation score for following
criteria
150     1 point for one 9 CFU OR 12 CFU.
151     1 point for two 6 CFU
152     Each point can be obtained once to total two point maximum.
153     */
154
155     for (int i = 0; i < numCourses; i++) { //runs each cell of array until final one
156         if ((courses[i].cfu == 9 || courses[i].cfu == 12) && courses[i].withLode == 1) {
157             // searches for credits 9 OR (||) 12 CFU with withLode argument 1
158             nineLodes++; //counts each time condition success
159             if (nineLodes == 1){ // catches 1st Lode possible
160                 totalLodes++; // increases total lode 1. We use increasing count rather than assign since
we use this argument below also
161             }
162         }
163         if ((courses[i].cfu == 6) && courses[i].withLode == 1) {
164             // searches for credits with 6 CFU with withLode argument 1
165             sixLodes++;
166             if (sixLodes == 2){// condition is TWO 6 CFU with lode.
167                 totalLodes++; // increases total lode 1. so if there is also 9 CFU Lode. total will be
maximum 2
168             }
169         }
170     }
171
172     // argument for calculating how much CFU is needed to graduate
173     int neededCFU = 0;
174     //Total CFU of Mechatronics is 120. Subtracting taken CFU's gives us required CFU
175     neededCFU = 120 - totalCFU;
176
177     //We dont use 120 but we use 108 since 12 credits is thesis and will considered later simulation.
178     if (totalCFU < 108){
179
180         // states required CFU to graduate.
181         printf("%d CFU needed for graduation \n ", neededCFU);
182
183         // returns total Lode bonus.
184         printf("\nYou have %d extra bonus for Lode \n ", totalLodes);
185
186     }
187
188     /* 108 is maximum CFU can be taken before Thesis since thesis is 12 CFU.
189        when totalCFU is 108 this means that user eligible to graduate after giving
190        Thesis. This sequence of algorithm calculates extra points of graduation with
191        thesis points to output graduation score, possible Cum Laude status.
```

```
192
193         If thesis is not given yet program gives output of required thesis note to:
194         graduate or if its possible; obtain Cum Laude.
195      */
196      if (totalCFU == 108){
197          printf("You completed all CFU can be obtained from exams. System will continue with thesis
calculation to obtain your Graduation Grade. ");
198
199          //graduation score with bonuses included
200          float GradAvg = 0;
201
202          //final weighted average with thesis score included
203          float finalAVG = 0;
204
205          //status of student if is in normal study year. we didnt use boolean since it has 1 point value
already.
206          int intime = 0;
207
208          //thesis score out of 7
209          int thesis = 0;
210
211          // constant thesis CFU is 12.
212          int thesisCFU = 0;
213
214          printf("\n ---------------------------------------- \n"); //d ivider
215
216          printf("Student graduated in time?: (1 for yes, 0 for no)");
217          scanf("%d", &intime); // memory access to asset point of 0-1
218
219          printf("Write score of Thesis: (out of 7. if not presented input 0) ");
220          scanf("%d", &thesis);
221          // Lode bonus
222          printf("you have total Lode Bonus: %d \n", totalLodes);
223
224          /* this condition is for thesis is not 0 means it presented successfully.
225          for this reason thesis CFU will be 12 since this CFU is obtained.
226          */
227          if (thesis != 0){
228              thesisCFU = 12; //thesis CFU is 12. We dont want to directly change memory data since we will
use it in other conditions.
229
230              //Final Average = Weighted Average*TotalCFU (108) + thesis (7 point) * 30/7 (conversion to
30)*thesisCFU / 120 CFU
231              finalAVG = ((weightedGrade*totalCFU) + (thesis*30*thesisCFU/7)) / (totalCFU + thesisCFU);
232              printf("Your final weighted average with thesis, excluding bonuses: %.2f  \n", finalAVG); //
.XX format float
233              // we use float because float uses 4 bytes of memory, double uses 8 byte of memory
234
235
236              //Graduation Score: WeightedAverage * 110/30 (conversion to 110) + Lode Bonus (2 max) +
Graduated in 2 year (1 point) + thesis (7 max)
237              GradAvg = finalAVG*110/30 + totalLodes + intime + thesis;
238              printf("Your final score with thesis, included bonuses: %.2f  \n \n", GradAvg);
239
240
241              /* gives status of Cum Laude with graduation score.
242                 In Mechatronics 110+ rewards as Cum Laude
243                 If student gets 104 or higher Board can give a Cum Laude to student
244                 If score is lower than 66 student cannot graduate with that Score
245              */
246
247              // Score between 104 and 109
248              if (GradAvg < 110 && GradAvg > 103){
249                  printf("You are eligible to graduate with Cum Laude by Board decision.\n");
250                  printf("Grade points higher than 104 MAY given Cum Laude with decision of Board.\n");
251              }
```

```
252             // Score higher than or equal to 110
253             if (GradAvg >= 110){
254                 printf("You are eligible to graduate with Cum Laude\n");
255             }
256 
257             // minimum grade for graduation is 66
258             if (GradAvg >= 66 && GradAvg < 104){
259                 printf("You are eligible to graduate\n");
260             }
261 
262             // Score lower than 66
263             if (GradAvg < 66){
264                 printf("You are not eligible to graduate\n");
265             }
266 
267         }
268         /* this condition is for thesis is 0 means it will present in future.
269             for this reason thesis CFU will be 0 since not obtained.
270             this section calculates approximate thesis score needed to get different statuses
271         */
272         if (thesis == 0){
273             // thesis CFU set to 0
274             thesisCFU = 0;
275             // final average is not changed.
276             finalAVG = weightedGrade;
277 
278             //Graduation Score: WeightedAverage * 110/30 (conversion to 110) + Lode Bonus (2 max) +
Graduated in 2 year (1 point) + thesis (7 max)
279             //thesis is not included since thesis score is 0.
280             GradAvg = finalAVG*110/30 + totalLodes + intime + thesis;
281 
282             // gives warning to the requirement of thesis and prints current graduation mark.
283             printf("You need to finish your thesis to graduate. Your graduation mark without thesis: %.2f
\n", GradAvg);
284             printf("\n ---------------------------------------- \n \n"); //divider
285 
286             // Score between 103 and 97 have possibility to reach or surpass 104 threshold for Board
decision
287             if ( GradAvg < 104 && GradAvg >= 97){
288                 printf("In order to graduate with Cum Laude by board decison you need minimum: %.2f thesis
score \n \n", 104 - GradAvg); //this gives minimum thesis score needs to be obtained
289                 printf("Grade points higher than or equal to 104 MAY given Cum Laude with decision of
Board.\n\n");
290             }
291 
292             // Score between 109 and 103 since thesis score can be maximum 7.
293             if (110 - GradAvg <= 7 && GradAvg < 110){
294                 printf("In order to graduate with Cum Laude with 110+ marking you need minimum: %.2f thesis
score \n \n", 110 - GradAvg);
295                 printf("Grade points higher than or equal to 104 MAY given Cum Laude with decision of
Board.\n\n");
296 
297             }
298             // Score between 59 and 65 since these scores are only scores can thesis surpass.
299             if (GradAvg >= 59 && GradAvg < 66){
300                 printf("In order to graduate you need minimum: %.2f thesis score \n \n", 66 - GradAvg);
301 
302             }
303 
304             // Scores between 66 to 96 which has zero possibility to reach 104
305             if (GradAvg >= 66 && GradAvg < 97){
306                 printf("you are eligible to graduate. \nNo possibility of Cum Laude detected regarding your
grade.\nMinimum 104 required for Cum Laude.\n");
307                 printf("maximum score you can get: %.2f  \n \n", GradAvg + 7);
308 
309             }
```

```
310             // scores which have zero possibility to pass 66 limit
311             if(GradAvg < 59){
312                 printf("you are not eligible to graduate.");
313 
314             }
315         }
316     }
317     return 0; // program terminated normally.
318 }
```