

# **CSE101 FINAL REPORT**

## TOPIC: CA EVALUATOR SYSTEM

**DESCRIPTION:** The CA evaluation system is a computer program designed to automate the process of evaluating the performance of students in a course over a period of time. This system is commonly used in educational institutions to measure the progress of students throughout a semester or academic year. The main aim of this project is to develop a CA evaluation system using the C programming language.

The primary objective of this project is to create a CA evaluation system that can automate the process of evaluating student performance in a course. The system will be designed to handle the following tasks:

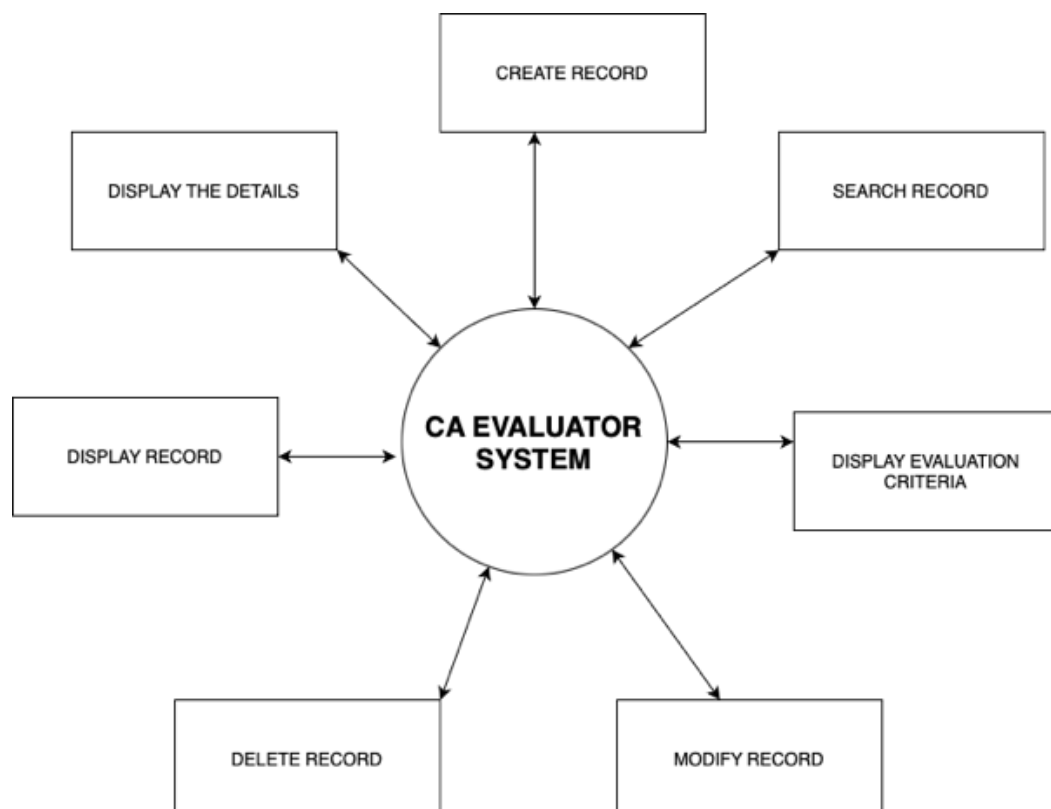
- Create Record
- Search Record
- Display Evaluation Criteria
- Modify Record
- Delete Record
- Display Record
- Display The Details

## MODULES EXPLANATION:

- **Create Record:** The Create Record module enables users to create a new student record in the system. This module allows users to input all the necessary information related to a student, such as their name, ID number, course, and other relevant details. Users can also input additional information such as the student's contact information, enrolment status, and any other relevant notes.
- **Search Record:** The Search Record module allows users to search for specific student records based on their name, ID number, or other relevant information. This module provides a search bar where users can enter the search criteria and find the desired student record. The search function is designed to be fast and efficient, enabling users to find the relevant records quickly.
- **Display Evaluation Criteria:** The Display Evaluation Criteria module enables users to view the evaluation criteria for each course in the system. Users can view the weightage of each component of the evaluation, such as assignments, quizzes, and final exams. This module provides users with a comprehensive overview of the evaluation process, enabling them to make informed decisions while evaluating student performance.

- **Modify Record:** The Modify Record module allows users to modify an existing student record in the system. Users can update the student details, change the evaluation scores, or add any other relevant information to the record. This module enables users to make any necessary changes to the student record without having to create a new record from scratch.
- **Delete Record:** The Delete Record module enables users to delete an existing student record from the system. This module is designed to be user-friendly and secure, ensuring that users can delete records without accidentally deleting the wrong record. Users can select the record they wish to delete and confirm the action before proceeding.
- **Display Record:** The Display Record module allows users to view all the student records in the system. Users can view the student details and their evaluation scores in a tabular format. This module provides users with an overview of all the student records, enabling them to identify any trends or patterns in student performance.
- **Display The Details:** The Display The Details module provides detailed information about each student record. Users can view the evaluation scores for each component of the evaluation, such as assignments, quizzes, and final exams. Additionally, users can view any notes or comments made by the evaluator. This module enables users to view all the relevant information related to a student record in one place.

DATA FLOW DIAGRAM (LEVEL 0):



## PROGRAMMING CODE:

```
CA EVALUATOR SYSTEM

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <time.h>
5
6 #define MAX_STUDENTS 1000
7 #define MAX_NAME_LENGTH 50
8 #define FILE_NAME "students_data.dat"
9 #define TEMP_FILE_NAME "temp.dat"
10 #define MAX_USERNAME_LENGTH 20
11 #define MAX_PASSWORD_LENGTH 20
12 #define DATABASE_FILE "database.dat"
13
14 // Structure to hold student record details
15 struct student
16 {
17     int id;
18     char name[MAX_NAME_LENGTH];
19     float att;
20     float ca;
21     float mte;
22     float ete;
23 };
24
25 struct user
26 {
27     char id[MAX_USERNAME_LENGTH];
28     char name[MAX_USERNAME_LENGTH];
29     char pass[MAX_PASSWORD_LENGTH];
30 };
31
32 void eva()
33 {
34     FILE *fp;
35     char c;
36
37     // Open the file
38     fp = fopen("evaluation.dat", "r");
39
40     // Check if the file exists
41     if (fp == NULL) {
42         printf("Unable to open file.\n");
43     }
44     printf("\n\nEvaluation Criteria\n\n");
45     // Read and print each character in the file
46     while ((c = fgetc(fp)) != EOF) {
47         putchar(c);
48     }
49
50     // Close the file
51     fclose(fp);
52     printf("\n\nPress any key to countinue.....");
53     getch();
54     return;
55 }
56
```

```

57 int random_id() {
58     int random_num;
59
60     // Seed the random number generator
61     srand(time(NULL));
62
63     // Generate a random number between 1000 and 9999
64     random_num = rand() % 9000 + 1000;
65     return random_num;
66 }
67
68 // Function to search for a student record by id
69 void see_detail(char *username) {
70     FILE *fp;
71     struct user u;
72     fp = fopen(DATABASE_FILE, "rb");
73     if (fp == NULL) {
74         printf("Failed to open file.\n");
75         return;
76     }
77     while (fscanf(fp, "%s %s %s\n", u.id, u.name, u.pass) != EOF) {
78         if (strcmp(username, u.id) == 0) {
79             printf("\n\nID: %s\nName: %s\nPassword: %s\n", u.id, u.name, u.pass);
80         }
81     }
82     fclose(fp);
83     printf("\n\nPress any key to countinue.....");
84     getch();
85     return;
86 }
87
88 void create_record() {
89     FILE *fp;
90     struct student s;
91     fp = fopen(FILE_NAME, "ab");
92     if (fp == NULL) {
93         printf("Failed to open file.\n");
94         return;
95     }
96     int id=random_id();
97     while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete)
98         != EOF){
99         if (s.id == id)
100             id=random_id();
101     }
102     s.id=id;
103     printf("Enter student name: ");
104     scanf("%s", s.name);
105     printf("Enter Attendane marks(out of 100): ");
106     scanf("%f", &s.att);
107     printf("Enter CA marks(out of 100): ");
108     scanf("%f", &s.ca);
109     printf("Enter Midterm marks(out of 100): ");
110     scanf("%f", &s.mte);
111     printf("Enter Endterm marks(out of 100): ");
112     scanf("%f", &s.ete);
113     fprintf(fp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca, s.mte,s.ete);
114     fclose(fp);
115     printf("Record created successfully.\n");
116     printf("\n\nPress any key to countinue.....");
117     getch();
118 }

```

```

119 // Function to search for a student record by id
120 void search_record() {
121     FILE *fp;
122     struct student s;
123     int search_id;
124     fp = fopen(FILE_NAME, "rb");
125     if (fp == NULL) {
126         printf("Failed to open file.\n");
127         return;
128     }
129     printf("Enter student id to search: ");
130     scanf("%d", &search_id);
131     while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete)
    != EOF) {
132         if (s.id == search_id) {
133             printf("ID: %d\nName: %s\nAttendance Marks: %.2f\nCA Marks: %.2f\nMidterm
    Marks: %.2f\nEndterm Marks: %.2f", s.id, s.name, s.att,s.ca,s.mte,s.ete);
134             fclose(fp);
135             printf("\n\nPress any key to countinue.....");
136             getch();
137             return;
138         }
139     }
140     printf("Record not found.\n");
141     fclose(fp);
142     printf("\n\nPress any key to countinue.....");
143     getch();
144 }
145
146 // Function to modify an existing student record by id
147 void modify_record() {
148     display_record_inner();
149     FILE *fp, *fp_temp;
150     struct student s;
151     int modify_id, found = 0;
152     fp = fopen(FILE_NAME, "rb");
153     fp_temp = fopen(TEMP_FILE_NAME, "wb");
154     if (fp == NULL || fp_temp == NULL) {
155         printf("Failed to open file.\n");
156         return;
157     }
158     printf("\nEnter student id to modify: ");
159     scanf("%d", &modify_id);
160     while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete)
    != EOF) {
161         if (s.id == modify_id) {
162             found = 1;
163             printf("Enter new attendance marks(out of 100): ");
164             scanf("%f", &s.att);
165             printf("Enter new CA marks(out of 100): ");
166             scanf("%f", &s.ca);
167             printf("Enter new Midterm marks(out of 100): ");
168             scanf("%f", &s.mte);
169             printf("Enter new Endterm marks(out of 100): ");
170             scanf("%f", &s.ete);
171             fprintf(fp_temp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca,
    s.mte,s.ete);
172             // fprintf(fp_temp, "%d %s %f\n", s.id, s.name, s.ca);
173             printf("Record modified successfully.\n");
174         } else {
175             fprintf(fp_temp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca,
    s.mte,s.ete);
176             //fprintf(fp_temp, "%d %s %f %f %f\n", s.id, s.name, s.ca1, s.ca2,s.ca3);
177             // fprintf(fp_temp, "%d %s %f\n", s.id, s.name, s.ca);
178         }
179     }
180     if (!found) {
181         printf("Record not found.\n");
182     }

```

```

183     fclose(fp);
184     fclose(fp_temp);
185     remove(FILE_NAME);
186     rename(TEMP_FILE_NAME, FILE_NAME);
187     printf("\n\nPress any key to countinue.....");
188     getch();
189 }
190
191 // Function to delete an existing student record by id
192 void delete_record() {
193     display_record_inner();
194     FILE *fp, *fp_temp;
195     struct student s;
196     int modify_id, found = 0;
197     fp = fopen(FILE_NAME, "rb");
198     fp_temp = fopen(TEMP_FILE_NAME, "wb");
199     if (fp == NULL || fp_temp == NULL) {
200         printf("Failed to open file.\n");
201         return;
202     }
203     printf("\nEnter student id to delete: ");
204     scanf("%d", &modify_id);
205     while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete)
206 != EOF) {
207         if (s.id == modify_id) {
208             found = 1;
209             printf("\nRecord Deleted Successfully\n");
210             continue;
211         } else {
212             fprintf(fp_temp, "%d %s %f %f %f %f\n", s.id, s.name, s.att, s.ca,
213 s.mte,s.ete);
214             // fprintf(fp_temp, "%d %s %f\n", s.id, s.name, s.ca);
215         }
216     }
217     if (!found) {
218         printf("Record not found.\n");
219     }
220     fclose(fp);
221     fclose(fp_temp);
222     remove(FILE_NAME);
223     rename(TEMP_FILE_NAME, FILE_NAME);
224     printf("\n\nPress any key to countinue.....");
225     getch();
226 }
227
228 void display_record_inner() {
229     FILE *fp;
230     struct student s;
231     float t=0.0;
232     char g[10];
233     char *gk = g;
234     fp = fopen(FILE_NAME, "rb");
235     if(fp == NULL) {
236         printf("Error in opening file\n");
237         return;
238     }
239     printf("\nID\tName\tAttendance Marks\tCA Marks\tMidterm Marks\tEndterm
240 Marks\tGrade\n");

```

```

239
240     while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte,&s.ete)
    != EOF) {
241         t=(s.att+s.ca+s.ete+s.mte)/(4.0);
242         if(t>=97 && t<=100)
243             gk="A+";
244         else if (t>=93 && t<=96)
245             gk="A";
246         else if (t>=90 && t<=92)
247             gk="A-";
248         else if (t>=87 && t<=89)
249             gk="B+";
250         else if (t>=83 && t<=86)
251             gk="B";
252         else if (t>=80 && t<=82)
253             gk="B-";
254         else if (t>=77 && t<=79)
255             gk="C+";
256         else if (t>=73 && t<=76)
257             gk="C";
258         else if (t>=70 && t<=72)
259             gk="C-";
260         else if (t>=67 && t<=69)
261             gk="D+";
262         else if (t>=65 && t<=66)
263             gk="D";
264         else if (t>=60 && t<=64)
265             gk="E";
266         else if (t<60)
267             gk="F";
268         printf("%d\t\t%s\t\t%.2f\t\t%.2f\t\t%.2f\t\t%.2f\t\t%s\n",s.id, s.name,
    s.att,s.ca,s.mte,s.ete,gk);
269     }
270     fclose(fp);
271 }
272
273 // Function to display the record
274 void display_record() {
275     FILE *fp;
276     struct student s;
277     float t=0.0;
278     char g[10];
279     char *gk = g;
280     fp = fopen(FILE_NAME, "rb");
281     if(fp == NULL) {
282         printf("Error in opening file\n");
283         return;
284     }
285
286     printf("\nID\t\tName\t\tAttendance Marks\tCA Marks\tMidterm Marks\tEndterm
    Marks\tGrade\n");
287

```

```

288     while (fscanf(fp, "%d %s %f %f %f %f\n", &s.id, s.name, &s.att, &s.ca, &s.mte, &s.ete)
    != EOF) {
289         t=(s.att+s.ca+s.ete+s.mte)/(4.0);
290         if(t>=97 && t<=100)
291             gk="A+";
292         else if (t>=93 && t<=96)
293             gk="A";
294         else if (t>=90 && t<=92)
295             gk="A-";
296         else if (t>=87 && t<=89)
297             gk="B+";
298         else if (t>=83 && t<=86)
299             gk="B";
300         else if (t>=80 && t<=82)
301             gk="B-";
302         else if (t>=77 && t<=79)
303             gk="C+";
304         else if (t>=73 && t<=76)
305             gk="C";
306         else if (t>=70 && t<=72)
307             gk="C-";
308         else if (t>=67 && t<=69)
309             gk="D+";
310         else if (t>=65 && t<=66)
311             gk="D";
312         else if (t>=60 && t<=64)
313             gk="E";
314         else if (t<60)
315             gk="F";
316         printf("%d\t\t%s\t\t%.2f\t\t%.2f\t\t%.2f\t\t%.2f\t\t%s\n", s.id, s.name,
    s.att, s.ca, s.mte, s.ete, gk);
317     }
318     fclose(fp);
319     printf("\n\nPress any key to countinue.....");
320     getch();
321 }
322
323 // Function to register a new user
324 void registerUser() {
325
326     char username[MAX_USERNAME_LENGTH];
327     char password[MAX_PASSWORD_LENGTH];
328     FILE *file;
329     file = fopen(DATABASE_FILE, "ab");
330     struct user u;
331     int found = 0;
332     if (file == NULL) {
333         printf("Failed to open file.\n");
334         return;
335     }
336     // printf("\nEnter Id: ");
337     // scanf("%d", &id);
338     printf("\nEnter your name (max %d characters): ", MAX_USERNAME_LENGTH);
339     scanf("%s", username);
340     printf("\nEnter a password (max %d characters): ", MAX_PASSWORD_LENGTH);
341     scanf("%s", password);
342     int num=random_id();
343     char str[MAX_USERNAME_LENGTH];
344     sprintf(str, "%d", num);
345     strcat(str, "@");
346     strcat(str, username);
347     fprintf(file, "%s %s %s\n", str, username, password);
348     printf("Your Username: %s\n", str);
349     printf("\nRegistration successful!\n");
350     printf("\n\nPress any key to countinue.....");
351     getch();
352     fclose(file);
353 }
354

```



```

355 // Function to check if a given username and password combination is valid
356 int checkCredentials(char *id, char *password) {
357     struct user u;
358     // Open the database file in read mode
359     FILE *file = fopen(DATABASE_FILE, "r");
360     if (file == NULL) {
361         printf("\nError: could not open database file\n");
362         exit(1);
363     }
364     // Loop through each line of the file
365     while (fscanf(file, "%s %s %s\n", u.id, u.name, u.pass) != EOF) {
366         // Check if the username and password match
367         if (strcmp(id, u.id) == 0 && strcmp(password, u.pass) == 0) {
368             fclose(file);
369             return 1;
370         }
371     }
372     fclose(file);
373     return 0;
374 }
375
376 // Function to handle the login process
377 void login() {
378     char id[MAX_USERNAME_LENGTH];
379     char username[MAX_USERNAME_LENGTH];
380     char password[MAX_PASSWORD_LENGTH];
381     struct user u;
382     printf("\nEnter your id: ");
383     scanf("%s", id);
384     printf("\nEnter your password: ");
385     scanf("%s", password);
386     if (checkCredentials(id, password)) {
387         printf("\nLogin successful!\n");
388         printf("\n\nPress any key to countinue.....");
389         getch();
390         FILE *file = fopen(DATABASE_FILE, "r");
391         if (file == NULL) {
392             printf("\nError: could not open database file\n");
393             exit(1);
394         }
395         while (fscanf(file, "%s %s %s\n", u.id, u.name, u.pass) != EOF){
396             if (strcmp(id, u.id) == 0)
397                 mainmenu(u.name, u.id);
398         }
399     } else {
400         printf("\nIncorrect id or username or password\n");
401         printf("\n\nPress any key to countinue.....");
402         getch();
403     }
404 }
405

```

```

406 int main() {
407     int choice;
408     do {
409         system("cls");
410         printf("\n*** CA Evaluator System ***\n");
411         printf("\tBy Rohan Chakravarty\n");
412         printf("*** Limitations ***\n");
413         printf("1. Maximun Name must be of 50 Characters Total\n");
414         printf("2. The System can hold 1000 records of the Student\n");
415         printf("3. May be Inconsistent and Redundant\n");
416         printf("\n\n*** MENU ***\n");
417         printf("1. New User/Register\n");
418         printf("2. Existing User/Login\n");
419         printf("3. Quit\n");
420         printf("Enter your choice: ");
421         scanf("%d", &choice);
422         switch (choice) {
423             case 1:
424                 registerUser();
425                 break;
426             case 2:
427                 login();
428                 break;
429             case 3:
430                 printf("\nThank you for Using the System\n");
431                 printf("\t~Rohan Chakravarty\n");
432                 printf("\n\nPress any key to countinue....");
433                 getch();
434                 exit(0);
435                 break;
436             default:
437                 printf("Invalid choice\n");
438                 break;
439         }
440     } while (choice != 3);
441     return 0;
442 }
443
444 void mainmenu(char str[], char str1[]) {
445     int choice;
446     while(1) {
447         system("cls");
448         printf("\nWelcome %s\n", str);
449         printf("*** MENU ***\n");
450         printf("1. Create Record\n");
451         printf("2. Search Record\n");
452         printf("3. Display Evaluation Criteria\n");
453         printf("4. Modify Record\n");
454         printf("5. Delete Record\n");
455         printf("6. Display Record\n");
456         printf("7. Display Your Detail\n");
457         printf("8. Logout\n");
458         printf("Enter your choice: ");
459         scanf("%d", &choice);
460

```

```
461     switch(choice) {
462         case 1:
463             create_record();
464             break;
465         case 2:
466             search_record();
467             break;
468         case 3:
469             eva();
470             break;
471         case 4:
472             modify_record();
473             break;
474         case 5:
475             delete_record();
476             break;
477         case 6:
478             display_record();
479             break;
480         case 7:
481             see_detail(str1);
482             break;
483         case 8:
484             return;
485             break;
486         default:
487             printf("Invalid choice\n");
488     }
489 }
490 }
```

## OUTPUT SNAPSHOT:

```
*** CA Evaluator System ***
      By Rohan Chakravarty
*** Limitations ***
1. Maximun Name must be of 50 Characters Total
2. The System can hold 1000 records of the Student
3. May be Inconsistent and Redundant

*** MENU ***
1. New User/Register
2. Existing User/Login
3. Quit
Enter your choice: 1

Enter your name (max 20 characters): Nishan

Enter a password (max 20 characters): nishan
Your Username: 9480@Nishan

Registration successful!

Press any key to countinue.....|

*** CA Evaluator System ***
      By Rohan Chakravarty
*** Limitations ***
1. Maximun Name must be of 50 Characters Total
2. The System can hold 1000 records of the Student
3. May be Inconsistent and Redundant

*** MENU ***
1. New User/Register
2. Existing User/Login
3. Quit
Enter your choice: 2

Enter your id: 9480@Nishan

Enter your password: nishan

Login successful!

Press any key to countinue.....|
```

```
Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 1
Enter student name: Nishan
Enter Attendance marks(out of 100): 97
Enter CA marks(out of 100): 95
Enter Midterm marks(out of 100): 91
Enter Endterm marks(out of 100): 98
Record created successfully.
```

Press any key to continue....|

```
Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 3
```

Evaluation Criteria

+-----+-----+	
Letter Grade	Percent Grade
+-----+-----+	
A+	97-100
A	93-96
A-	90-92
B+	87-89
B	83-86
B-	80-82
C+	77-79
C	73-76
C-	70-72
D+	67-69
D	65-66
E	60-64
F	Below 60
+-----+-----+	

Press any key to continue....|

```
Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 6

ID          Name          Attendance Marks    CA Marks    Midterm Marks    Endterm Marks    Grade
5439        Dona          100.00             100.00      100.00           100.00          A+
5471        Esha          4.00               4.00        4.00             4.00            F
5494        Samya         7.00               7.00        7.00             7.00            F
7251        Rohan         100.00             100.00      100.00           100.00          A+
1342        Nishan        97.00              95.00       91.00            90.00           A
```

Press any key to countinue.....|

```
Welcome Nishan
*** MENU ***
1. Create Record
2. Search Record
3. Display Evaluation Criteria
4. Modify Record
5. Delete Record
6. Display Record
7. Display Your Detail
8. Logout
Enter your choice: 7
```

```
ID: 9480@Nishan
Name: Nishan
Password: nishan
```

Press any key to countinue.....|