Main C

|  |
| --- |
| StartAlgorithm: integer main () |
|  |  |
|  |  |
|  | exit 🡨 0 |
|  | option 🡨 0 |
|  |  |
|  | while ( exit 🡨 0) do |
|  |  |
|  | char decision 🡨 0 |
|  | option 🡨 menu() |
|  | DoCase (option) |
|  |  |
|  | case 1 : addClass() |
|  | break |
|  | case 2 : removeClass() |
|  | break |
|  | case 3 : getClassRpt() |
|  | break |
|  | case 4 : mngClass() |
|  | break |
|  | case 6 : break |
|  | EndCase |
|  |  |
|  | Write("Would you like to terminate the program ? y/n ") |
|  | fflush(stdin) |
|  | Read (decision) |
|  |  |
|  | if ( decision 🡨 “y”) |
|  | exit 🡨 1 |
|  | else if (decision 🡨 “n”) |
|  | exit 🡨 0 |
|  | else |
|  | EndIf |
|  |  |
|  | Write("Sorry the input is not regonised, try using lower case, not upper case letters") |
|  | Write("Would you like to terminate the program ? y/n ") |
|  | fflush(stdin) |
|  | Read (decision) |
|  |  |
|  | if ( decision 🡨 ”y”) |
|  | exit 🡨 1 |
|  | else if (decision 🡨 ”n”) |
|  | exit 🡨 0 |
|  | EndIf |
|  | Write (NEWLINE) |
|  | EndWhile |
|  | Stop |

SCREEN.H

|  |
| --- |
|  |
| #include "utils.h" |
|  | #define NEWLINE "\n" |
|  |  |
|  | extern FILE \*ptr |
|  |  |
|  | menu() |
|  | Start |
|  | option 🡨 0 |
|  |  |
|  | Write("Please chose the number that corresponds with the actions you will like to commit :") |
|  | Write(NEWLINE) |
|  | Write("1) Add a class.") |
|  | Write(NEWLINE) |
|  | Write("2) Remove a class") |
|  | Write(NEWLINE) |
|  | Write("3) Get a class report") |
|  | Write(NEWLINE) |
|  | Write("4) Manage class") |
|  | Write(NEWLINE) |
|  | Write("6) Undecided") |
|  | Write(NEWLINE) |
|  |  |
|  | flushBuffer() |
|  | Write("What do wish to do : ") |
|  | Read(option) |
|  |  |
|  | return option |
|  | Stop |
|  |  |
|  | Start Algorithm: void addClass() |
|  |  |
|  | char cName[40] |
|  | char tName[40] |
|  | char tPass[40] |
|  |  |
|  | Write(NEWLINE) |
|  | Write("|\t|\t|\t|\t Add Class \t|\t|\t|\t|") |
|  | Write(NEWLINE) |
|  |  |
|  | Write("Please enter the class's name : ") |
|  | flushBuffer() |
|  | gets(cName) |
|  |  |
|  | Write("Please enter username : ") |
|  | flushBuffer() |
|  | gets(tName) |
|  |  |
|  | Write ("Please enter password for the class : ") |
|  | flushBuffer() |
|  | gets(tPass) |
|  |  |
|  | createClass(cName,tName,tPass) |
|  | Stop Algorithm |
|  |  |
|  | Start Algorithm: void removeClass() |
|  |  |
|  | char className[40] |
|  | char tUsername[40] |
|  | char tPassword[40] |
|  |  |
|  | Write ("Please enter the name of the class you'd like to delete : ") |
|  | flushBuffer() |
|  | gets(className) |
|  |  |
|  | strcat(className,".txt") |
|  |  |
|  | FILE \*fp |
|  | fp 🡨 fopen(className,"r") |
|  | If (!fp) then |
|  |  |
|  | Write("Class not found",className) |
|  | fclose(fp) |
|  |  |
|  | EndIF |
|  |  |
|  | Write("Please enter username : ") |
|  | flushBuffer() |
|  | gets(tUsername) |
|  |  |
|  | Write("Please enter password for class : ") |
|  | flushBuffer() |
|  | gets(tPassword) |
|  |  |
|  |  |
|  | If (fp) then |
|  |  |
|  | char readUsername[40] |
|  | char readPassword[40] |
|  | fscanf(fp,"%s\t%s",readUsername,readPassword) |
|  | fclose(fp) |
|  | if (strcmp(readPassword,tPassword) 🡨 0 && |
|  | strcmp(readUsername,tUsername) 🡨 0) |
|  |  |
|  | status 🡨 remove(className) |
|  | Write ("Status :",status) |
|  | else |
|  | Write ("User name and/or password is incorrect") |
|  | else |
|  |  |
|  | Write ("File not found",className) |
|  | EndIf |
|  |  |
|  | EndIf |
|  |  |
|  | Start Algorithm: void getClassRpt() |
|  |  |
|  | char tUsername[40] |
|  | char tPassword[40] |
|  | char className[40] |
|  |  |
|  | Write (NEWLINE) |
|  | Write ("|\t|\t|\t|\t Class Report \t|\t|\t|\t|") |
|  | Write (NEWLINE) |
|  | Write (NEWLINE) |
|  |  |
|  | Write ("Please enter class you'd like to get report from : ") |
|  | flushBuffer() |
|  | gets(className) |
|  |  |
|  | strcat(className,".txt") |
|  | FILE \*fp 🡨 fopen(className,"r") |
|  |  |
|  | Write ("Please enter usename : ") |
|  | flushBuffer() |
|  | gets(tUsername) |
|  |  |
|  | Write ("Please enter password : ") |
|  | flushBuffer() |
|  | gets(tPassword) |
|  |  |
|  | bool isLogged 🡨 loginTeacher(tUsername,tPassword,fp) |
|  |  |
|  | struct Student students[100] |
|  |  |
|  | If (isLogged) then |
|  |  |
|  |  |
|  | x 🡨 0 |
|  | i 🡨 0 |
|  |  |
|  | char heighestAverageStudent[40] |
|  | heighestAverage 🡨 0 |
|  |  |
|  | char heighestEngStudent[40] |
|  | heighestEng 🡨 0 |
|  |  |
|  | char heighestPhysEdStudent[40] |
|  | heighestPhyEd 🡨 0 |
|  |  |
|  | char heighestMathStudent[40] |
|  | heighestMath 🡨 0 |
|  |  |
|  | char heighestInteSciStudent[40] |
|  | heighestInteSci 🡨 0 |
|  |  |
|  | While (x ! 🡨 EOF) |
|  |  |
|  | struct Student student |
|  | x 🡨 fscanf(fp,"%s\t%d\t%d\t%d\t%d\n",student.fullName,&student.mathGrade,&student.engGrade,&student.phyEdGrade,&student.inteSciGrade) |
|  |  |
|  |  |
|  | strcpy(students[i].fullName,student.fullName) |
|  | students[i].mathGrade 🡨 student.mathGrade |
|  | students[i].engGrade 🡨 student.engGrade |
|  | students[i].phyEdGrade 🡨 student.phyEdGrade |
|  | students[i].inteSciGrade 🡨 student.inteSciGrade |
|  |  |
|  |  |
|  | If (students[i].mathGrade > heighestMath) then |
|  |  |
|  | heighestMath 🡨 students[i].mathGrade |
|  | strcpy(heighestMathStudent,students[i].fullName) |
|  | EndIf |
|  |  |
|  | if (students[i].engGrade > heighestEng) then |
|  |  |
|  | heighestEng 🡨 students[i].engGrade |
|  | strcpy(heighestEngStudent,students[i].fullName) |
|  | EndIf |
|  |  |
|  | if (students[i].phyEdGrade > heighestPhyEd) then |
|  |  |
|  | heighestPhyEd 🡨 students[i].phyEdGrade |
|  | strcpy(heighestPhysEdStudent,students[i].fullName) |
|  | EndIf |
|  |  |
|  | if (students[i].inteSciGrade > heighestInteSci) then |
|  |  |
|  | heighestInteSci 🡨 students[i].inteSciGrade |
|  | strcpy(heighestInteSciStudent,students[i].fullName) |
|  | EndIf |
|  |  |
|  | i++ |
|  | EndWhile |
|  |  |
|  | i 🡨 i-1 |
|  | j 🡨 0 |
|  | Write("Student Name/Math Average Grade/English Average Grade/Integrated Science Average Grade/Physical Education Average Grade/Overall Average") |
|  | while ( j < i) |
|  |  |
|  | averageGrade 🡨 (students[j].mathGrade + students[j].engGrade + students[j].inteSciGrade + students[j].phyEdGrade)/4 |
|  | Write("",students[j].fullName,students[j].mathGrade,students[j].engGrade,students[j].inteSciGrade,students[j].phyEdGrade,averageGrade) |
|  | j++ |
|  | EndWhile |
|  |  |
|  | Write("Heighest average in mathematics ",heighestMath,heighestMathStudent) |
|  | Write("Heighest average in english",heighestEng,heighestEngStudent) |
|  | Write("Heighest average in integrated science  ",heighestInteSci,heighestInteSciStudent) |
|  | Write("Heighest average in physical education ",heighestPhyEd,heighestPhysEdStudent) |
|  |  |
|  | EndIf |
|  |  |
|  | Stop Algorithm |
|  |  |
|  | Start Algorithm: void mngClass() |
|  |  |
|  | option 🡨 0 |
|  |  |
|  | Write (NEWLINE) |
|  | Write ("|\t|\t|\t|\t Manage Class \t|\t|\t|\t|") |
|  | Write (NEWLINE) |
|  | Write (NEWLINE) |
|  |  |
|  | Write ("Please chose the number that corresponds with the actions you will like to commit :") |
|  | Write (NEWLINE) |
|  | Write (NEWLINE) |
|  |  |
|  | Write ("1) Add Student to a class.") |
|  | Write (NEWLINE) |
|  |  |
|  | Write ("2) Remove Student from a class.") |
|  | Write (NEWLINE) |
|  |  |
|  | Write ("What would like to do : ") |
|  | Read(option) |
|  |  |
|  | if (option 🡨 1) then |
|  | addStudent() |
|  | else if(option 🡨 2) |
|  | removeStudent() |
|  | else |
|  |  |
|  | Write ("The value you entered is not recognised") |
|  | Write (NEWLINE) |
|  | return |
|  | EndIf |
|  | Stop Algorithm  UTILS.H   |  |  | | --- | --- | |  |  | |  | void addStudents(FILE \*) | |  |  | |  |  | |  | Start Algorithm: void flushBuffer() | |  |  | |  | fflush(stdin) | |  | Stop Algorithm | |  |  | |  |  | |  |  | |  |  | |  | Start Algorithm: void createClass(char cName[60], char tName[40],char tPass[40]) | |  |  | |  | FILE \*ptr | |  |  | |  | /\*\*strcat(location,cName) // concatenate the string path to the file name | |  | strcat(location,".txt") | |  | strcpy(path,location)\*\*/ | |  |  | |  | char \*fileExtension 🡨 ".txt" | |  |  | |  | strcat(cName,fileExtension) | |  | ptr 🡨 fopen(cName,"w") | |  |  | |  | // Error handling | |  | If (ptr) then | |  |  | |  | fprintf(ptr,"%s\t%s\n",tName,tPass) | |  | addStudents(ptr) | |  | Write("Class Created successfully") | |  | fclose(ptr) | |  | else | |  |  | |  | Write (NEWLINE) | |  | perror("Error") | |  | Write (NEWLINE) | |  | exit(1) | |  | EndIf | |  | Stop Algorithm | |  |  | |  | Start Algorithm: bool loginTeacher(char tName[40],char tPass[40],FILE \*fp) | |  |  | |  | // copied from removeClass function | |  | If (fp) then | |  |  | |  | char readUsername[40] | |  | char readPassword[40] | |  | fscanf(fp,"%s\t%s\n",readUsername,readPassword) | |  | //fclose(fp) | |  | if (strcmp(readPassword,tPass) 🡨 0 && | |  | strcmp(readUsername,tName) 🡨 0) then | |  |  | |  | return true | |  | else | |  |  | |  | Write ("User name and/or password is incorrect") | |  | return false | |  |  | |  | else | |  |  | |  | return false | |  | EndIf | |  | EndIf | |  |  | |  |  | |  | Start Algorithm: void addStudents(FILE \*fp) | |  |  | |  | size 🡨 0 | |  | flushBuffer() | |  | Write ("Please enter the amount of students that are in the class :") | |  | Read(size) | |  | struct Student students[size] | |  |  | |  |  | |  | for (i 🡨 0 i < size) | |  |  | |  | x 🡨 i + 1 | |  | Write ("Please enter student full name : ",x) | |  | flushBuffer() | |  | gets(students[i].fullName) | |  |  | |  | Write ("Please enter student Mathematics average grade : ",x) | |  | flushBuffer() | |  | Read (students[i].mathGrade) | |  |  | |  | Write ("Please enter student English average grade : ",x) | |  | flushBuffer() | |  | Read (students[i].engGrade) | |  |  | |  | Write ("Please enter student Physical Education average grade : ",x) | |  | flushBuffer() | |  | Read (students[i].phyEdGrade) | |  |  | |  | Write ("Please enter student Integrated Science average grade : ",x) | |  | flushBuffer() | |  | Read (students[i].inteSciGrade) | |  | EndFor | |  |  | |  | for ( i 🡨 0 i < size) | |  |  | |  | fprintf(fp,"%s\t%d\t%d\t%d\t%d\n",students[i].fullName,students[i].mathGrade,students[i].engGrade,students[i].phyEdGrade,students[i].inteSciGrade) | |  | EndFor | |  | Stop Algorithm | |  |  | |  | Start Algorithm: void addStudent() | |  |  | |  | char tUsername[40] | |  | char tPass[40] | |  | char className[40] | |  |  | |  | Write ("Please enter the class you'd like to add the student to : ") | |  | flushBuffer() | |  | gets(className) | |  |  | |  | strcat(className,".txt") | |  | FILE \*fp 🡨 fopen(className,"r") | |  |  | |  | Write ("Please enter username : ") | |  | flushBuffer() | |  | gets(tUsername) | |  |  | |  | Write ("Please enter password : ") | |  | flushBuffer() | |  | gets(tPass) | |  |  | |  | bool isLogged 🡨 loginTeacher(tUsername,tPass,fp) | |  |  | |  | fp 🡨 fopen(className,"a") | |  |  | |  | If (isLogged) then | |  |  | |  | struct Student student | |  | Write ("Please enter student full name : ") | |  | flushBuffer() | |  | gets(student.fullName) | |  |  | |  | Write ("Please enter student Mathematics average grade : ") | |  | flushBuffer() | |  | Read (student.mathGrade) | |  |  | |  | Write ("Please enter student English average grade : ") | |  | flushBuffer() | |  | Read (student.engGrade) | |  |  | |  | Write ("Please enter student Physical Education average grade : ") | |  | flushBuffer() | |  | Read (student.phyEdGrade) | |  |  | |  | Write ("Please enter student Integrated Science average grade : ") | |  | flushBuffer() | |  | Read (student.inteSciGrade) | |  | fprintf(fp,"%s\t%d\t%d\t%d\t%d\n",student.fullName,student.mathGrade,student.engGrade,student.phyEdGrade,student.inteSciGrade) | |  | fclose(fp) | |  |  | |  | Write ("Student added successfully") | |  | EndIf | |  | Stop Algorithm | |  |  | |  | Start Algorithm: void removeStudent() | |  |  | |  | char tUsername[40] | |  | char tPass[40] | |  | char className[40] | |  | char studentName[40] | |  |  | |  | Write ("Please enter the class you'd like to add the student to : ") | |  | flushBuffer() | |  | gets(className) | |  |  | |  | strcat(className,".txt"); // add .txt to the file extension | |  | FILE \*fp 🡨 fopen(className,"r") | |  |  | |  | Write ("Please enter username : ") | |  | flushBuffer() | |  | gets(tUsername) | |  |  | |  | Write ("Please enter password : ") | |  | flushBuffer() | |  | gets(tPass) | |  |  | |  | Write ("Please enter student that you'd like to remove : ") | |  | flushBuffer() | |  | gets(studentName) | |  |  | |  | struct Student students[100] | |  |  | |  | bool isLogged 🡨 loginTeacher(tUsername,tPass,fp) | |  |  | |  | If (isLogged) then | |  |  | |  | x 🡨 1 | |  | i 🡨 0 | |  |  | |  | struct Student student | |  |  | |  | while (x ! 🡨 EOF) | |  |  | |  | x 🡨fscanf(fp,"%s\t%d\t%d\t%d\t%d\n",student.fullName,&student.mathGrade,&student.engGrade,&student.phyEdGrade,&student.inteSciGrade) | |  | Write ("Name : i 🡨",student.fullName,i) | |  |  | |  | strcpy(students[i].fullName,student.fullName) | |  | students[i].mathGrade 🡨 student.mathGrade | |  | students[i].engGrade 🡨 student.engGrade | |  | students[i].phyEdGrade 🡨 student.phyEdGrade | |  | students[i].inteSciGrade 🡨 student.inteSciGrade | |  | i++ | |  | EndWhile | |  |  | |  |  | |  | fclose(fp) | |  | remove(className) | |  | j 🡨 i | |  |  | |  | FILE \*ptr 🡨 fopen(className,"w") | |  | fprintf(ptr,"%s\t%s",tUsername,tPass) | |  |  | |  | length 🡨 sizeof(students) / sizeof(students[0]) | |  | Write (" j :",j) | |  |  | |  | for ( i 🡨 0; i < length) | |  |  | |  | if ( strcmp(studentName,students[i].fullName) 🡨 0) | |  | continue | |  | else if ( i 🡨 j) | |  | break | |  |  | |  | fprintf(ptr,"%s\t%d\t%d\t%d\t%d\n",students[i].fullName,students[i].mathGrade,students[i].engGrade,students[i].phyEdGrade,students[i].inteSciGrade) | |  | EndFor | |  |  | |  | EndIf | |  | Stop Algorithm | |
|  |  |
|  |  |