INTRODUCTION:

This C++ program is designed to efficiently manage student assignments within university dormitories. It provides robust functionality to accept new student data, assign them to dormitory rooms, delete student records, find specific student details, and update existing information. By leveraging advanced data structures and algorithms, this program ensures smooth and accurate handling of student records, making dormitory management streamlined and hassle-free. Whether you're adding new students, updating their dorm assignments, or retrieving specific information, this program is a comprehensive solution tailored to the unique needs of university dormitory administration.

THE PROGRAM:

```
#include <iostream>
#include <string.h>
#include <cstdlib>
#include <fstream>
#include <conio.h>
using namespace std;
struct newstud {
int roomnum;
char blocktype;
char name[50];
char id[10];
char gender[10];
char department[50];
char cgpa[10];
};
void insertStudent();
void update();
void displayStudents();
void findStudents();
void deleteStudents();
void assignByName();
void assignByCgpa();
void displayDorms();
void summary();
void menu();
void about();
int main(){
```

```
ofstream check("newstuds.txt",ios::app);
if(check.fail()){
cerr<<"\t\t\n404!!error running program\n";
exit(1);
}
check.close();
system("color 7D");
char key;
cout <<"\n\n\t
welcome to Hawassa iot dormitories
-\n";
cout<<"\t\t\tenter any key to continue\n";
cout<<"\t\t\t\tenter x to exit\n";
cin>>key;
switch(key){
case 'x':
system("cls");
cout << "
---" << endl;
cout << "
THANK YOU
" << endl;
cout << "
FOR ACCESSING IOT DORMITORIES
" << endl;
cout << "
---" << endl;
cout << "
-" << endl << "
-" << endl << "
-" <<
endl << "
-" << endl;
getch();
cout << "
BYE\n";
cout<<"===========exited succesfully==========";
break;
default:
system("cls");
menu();
}
return 0;
}
void insertStudent(){
int marginal;
newstud students;
cout<<"\n\n\thow many students\n";
cin>>marginal;
ofstream registerstuds("newstuds.txt",ios::app);
for(int i=0; i<marginal; i++){
cout << "Enter student
name: ";
cin.ignore();
```

```
cin.getline(students.department,50);
cout << "Enter student CGPA: ";
cin >> students.cgpa;
students.roomnum=0;
if(strcmp(students.gender, "Male")==0)
students.blocktype='M';
else if(strcmp(students.gender, "Female")==0)
students.blocktype='F';
registerstuds.write((char*)&students, sizeof(newstud));
system("cls");
}
cout << "Student inserted successfully." << endl;
registerstuds.close();
}
void update(){
newstud students;
int total=0;
ifstream countStuds("newstuds.txt");
while(countStuds.read((char*)&students, sizeof(newstud)))
total++;
countStuds.close();
ifstream liststuds("newstuds.txt");
if(total==0){
cout<<"\n\n\t---no students available------\n";
}
else{
bool found=false;
string k;
cout<<"enter name or id to update"<<endl;
cin.ignore();
getline(cin,k);
fstream file("newstuds.txt",ios::in|ios::out);
if(!file){
cerr<<"error can't open\n";
exit(1);
}
while(!file.eof()&&found==false){
file.read((char*)&students, sizeof(newstud));
if(students.name==k||students.id==k){
found=true;
int x;
do{
```

```
cout <<"choose from the following\n";
cout<<"1: update name\n";
cout<<"2: update id\n";
cout<<"3: update gender\n";
cout<<"4: update department\n";
cout<<"5: update CGPA\n";
cout<<"6: done\n";
cin>>x;
system("cls");
switch (x){
case 1:
cout<<" old name= "<<students.name<<" \nenter the new\n";
cout<<"name: ";
cin.ignore();
cin.getline(students.name, 50);
system("cls");
break;
case 2:
cout<<" old id= "<<students.id<<" enter the new\n";
cout<<"
";
cin>>students.id;
system("cls");
break;
case 3:
cout<<" old gender= "<<students.gender<<" enter the new\n";
cout<<"gender: ";
cin>>students.gender;
students.gender[0]=toupper(students.gender[0]);
if(strcmp(students.gender,"Male")==0)
students.blocktype='M';
else
students.blocktype='F';
system("cls");
break;
case 4:
cout<<" old department= "<<students.department<<" enter the
new\n";
cout<<"department: ";
cin.ignore();
cin.getline(students.department,50);
system("cls");
break;
case 5:
cout<<" old cgpa= "<<students.cgpa<<" enter the new\n";
cout<<"CGPA: ";
cin>>students.cgpa;
system("cls");
break;
case 6:
cout<<"\n\t////updated succesfully////\n";
int pos=(-1)*static_cast<int>(sizeof(students));
file.seekp(pos,ios::cur);
file.write((char*)&students,sizeof(newstud));
```

```
}while(x!=6);
return;
}
}
cout<<"\n\n\tno such a student\n";
file.close();
cout<<"\n\tclick any key to continue\n";
getch();
system("cls");
}
}
void displayStudents(){
newstud students;
int total=0;
ifstream countStuds("newstuds.txt");
if(countStuds.fail()){
cerr<<"\n\n\tsorry!! error opening file\n";
exit(1);
}
while(countStuds.read((char*)&students, sizeof(newstud)))
total++;
countStuds.close();
ifstream listStuds("newstuds.txt");
if(listStuds.fail()){
cerr<<"\n\n\tsorry!! error opening file\n";
exit(1);
}
if(total==0){
cout<<"\n\n\t---no students available------\n";
}
else{
int num=1;
cout<<"\n\t"<<total<<" students\n";
while(listStuds.read((char*)&students, sizeof(newstud))){
cerr<<"\nstudent "<<num<<endl;
cout<<"name:- " << students.name << ", "
<< "Id number:- "<< students.id << ", "
<<"Gender:- "<< students.gender << ", "
<<"department:- "<< students.department
<< ", "<<"Cgpa:- " << students.cgpa<<", ";
if(students.roomnum==0)
cout<<" dorm:- unassigned yet, ";
else
cout<<" dorm:- "<<students.roomnum<<", ";
cout<<" block "<<students.blocktype<<endl;
num++;
}
listStuds.close();
```

```
}
cout<<"\n\tclick any key to continue\n";
getch();
system("cls");
}
void findStudent(){
int total=0;
bool found=false;
newstud students;
int x;
ifstream countStuds("newstuds.txt");
if(countStuds.fail()){
cerr<<"\n\n\tsorry!! error opening file\n";
exit(1);
}
while(countStuds.read((char*)&students, sizeof(newstud)))
total++;
countStuds.close();
if(total==0){
cout<<"\n\n\t---no students available------\n";
}
else{
do
{
cout<<"find the student based on:-\n"
<<"1: name\n"
<<"2: id\n"
<<"or press 3 to leave\n";
cin>>x;
system("cls");
switch(x){
case 1:{
ifstream namefile("newstuds.txt");
string name;
cout << "Enter student name: ";
cin >> name;
while(namefile.read((char*)&students, sizeof(newstud))) {
if (students.name==name) {
cout << "Student found:" << endl;
cout <<"name:- " <<students.name << endl << "id number:- "<<students.id << endl
<<"gender:- "<< students.gender
<< endl<<"department:- "<< students.department << endl<<"CGPA:- "
<< students.cgpa <<endl<<"block "<<students.blocktype<<endl;
if(students.roomnum==0){
cout<<"dorm:- unassigned yet\n";
}
else
cout<<" dorm:-"<<students.roomnum<<endl;
```

```
found=true;
break;
}
}
namefile.close();
if(found==false)
cout << "Student not found....." << endl;
break;
}
case 2:{
ifstream idfile("newstuds.txt");
string id;
cout << "Enter student
";
cin >> id;
while(idfile.read((char*)&students, sizeof(newstud))) {
if (students.id==id) {
cout << "\t Student found:\t" << endl;
cout <<"name:- " <<students.name << endl << "id number:- "<<students.id << endl
<<"gender:- "<< students.gender
<< endl<<"department:- "<< students.department << endl<<"CGPA:- " <<
students.cgpa <<endl<<"block "<<students.blocktype<<endl;
if(students.roomnum==0){
cout<<"dorm:- unassigned yet\n";
}
else
cout<<" dorm:-"<<students.roomnum<<endl;
found=true;
break;
}
}
idfile.close();
if(found==false)
cout << "Student not found....." << endl;
break;
}
case 3:
break;
default:
cout<<"\n\nerror!!!\n";
}
}while(x!=3);
cout<<"\n\tclick any key to continue\n";
getch();
system("cls");
}
}
void deleteStudent(){
newstud students;
char name[50];
cout << "Enter student name: ";
```

```
cin >> name;
ifstream infile("newstuds.txt");
ofstream outfile("newdorm.txt",ios::app);
int c=0;
bool found=false;
char ans;
while(infile.read((char*)&students, sizeof(newstud))) {
if (strcmp(students.name, name) != 0) {
outfile.write((char*)&students, sizeof(newstud));
}
else if(strcmp(students.name, name)==0){
found=true;
cout<<endl<<students.name<<endl<<"gender= "<<students.gender<<endl<<"department=
"<<students.department<<endl;
cout<<"\n\t\tare you sure you want to delete the above student?(Y/N)\n";
cin>>ans;
ans=toupper(ans);
if(ans!='Y'){
outfile.write((char*)&students, sizeof(newstud));
}
else
c++;
}
}
if(!found){
cout << "Student not found." << endl;
}
else if(c!=0){
cout << "Student deleted successfully." <<endl;
}
infile.close();
outfile.close();
remove("newstuds.txt");
rename("newdorm.txt", "newstuds.txt");
}
void assignByName(){
//count bzatachewn
int total=0;
newstud y;
ifstream countstud("newstuds.txt");
while(countstud.read((char*)&y, sizeof(newstud)))
total++;
countstud.close();
//copy to the array
newstud x[total];
int i=0;
ifstream copystud("newstuds.txt");
while(copystud.read((char*)&y, sizeof(newstud))){
x[i]=y;
i++;
```

```
}
copystud.close();
for (int i = 0; i < total - 1; ++i) {
for (int j = 0; j < total - 1 - i; ++j) {
if (strcmp(x[j].name, x[j + 1].name) > 0) {
// Swap the students if they are in the wrong order
newstud temp = x[j];
x[j] = x[j + 1];
x[j + 1] = temp;
}
}
}
newstud femalex[total],malex[total];
int findex=0,mindex=0;
for(int i=0;i<total;i++){
if(strcmp(x[i].gender, "Male")==0){
malex[mindex]=x[i];
mindex++;
}
else if(strcmp(x[i].gender, "Female")==0){
femalex[findex]=x[i];
findex++;
}
}
ofstream insertstuds("newstuds.txt");
int roomnom=1,roomnof=1;
for(int i=0;i<mindex;i++){
if(i%2==0){
malex[i].roomnum=roomnom;
insertstuds.write((char*)&malex[i], sizeof(newstud));
}
else{
malex[i].roomnum=roomnom;
insertstuds.write((char*)&malex[i], sizeof(newstud));
roomnom++;
}
}
for(int i=0;i<findex;i++){
if(i%2==0){
femalex[i].roomnum=roomnof;
insertstuds.write((char*)&femalex[i], sizeof(newstud));
}
else{
femalex[i].roomnum=roomnof;
insertstuds.write((char*)&femalex[i], sizeof(newstud));
roomnof++;
}
}
insertstuds.close();
}
void assignByCgpa(){
```

```
int total=0;
newstud y;
ifstream countstud("newstuds.txt");
while(countstud.read((char*)&y, sizeof(newstud)))
total++;
countstud.close();
//copy to the array
newstud x[total];
int i=0;
ifstream copystud("newstuds.txt");
while(copystud.read((char*)&y, sizeof(newstud))){
x[i]=y;
i++;
}
copystud.close();
for (int i = 0; i < total - 1; ++i) {
for (int j = 0; j < total - 1 - i; ++j) {
if (strcmp(x[j].cgpa, x[j + 1].cgpa) > 0) {
// Swap the students if they are in the wrong order
newstud temp = x[j];
x[j] = x[j + 1];
x[j + 1] = temp;
}
}
}
newstud femalex[total],malex[total];
int findex=0,mindex=0;
for(int i=0;i<total;i++){
if(strcmp(x[i].gender, "Male")==0){
malex[mindex]=x[i];
mindex++;
}
else if(strcmp(x[i].gender, "Female")==0){
femalex[findex]=x[i];
findex++;
}
}
ofstream insertstuds("newstuds.txt");
int roomnom=1,roomnof=1;
for(int i=0;i<mindex;i++){
if(i%2==0){
malex[i].roomnum=roomnom;
insertstuds.write((char*)&malex[i], sizeof(newstud));
}
else{
malex[i].roomnum=roomnom;
insertstuds.write((char*)&malex[i], sizeof(newstud));
roomnom++;
}
}
```

```
for(int i=0;i<findex;i++){
if(i%2==0){
femalex[i].roomnum=roomnof;
insertstuds.write((char*)&femalex[i], sizeof(newstud));
}
else{
femalex[i].roomnum=roomnof;
insertstuds.write((char*)&femalex[i], sizeof(newstud));
roomnof++;
}
}
insertstuds.close();
}
void displayDorms(){
int total=0;
newstud y;
ifstream countstud("newstuds.txt");
while(countstud.read((char*)&y, sizeof(newstud)))
total++;
countstud.close();
newstud x[total];
int i=0;
ifstream copystud("newstuds.txt");
while(copystud.read((char*)&y, sizeof(newstud))){
x[i]=y;
i++;
}
copystud.close();
int findex=0,mindex=0;
for(int i=0;i<total;i++){
if(strcmp(x[i].gender, "Male")==0){
mindex++;
}
else if(strcmp(x[i].gender, "Female")==0){
findex++;
}
}
newstud lads;
ifstream inFileRooms("newstuds.txt");
if (!inFileRooms) {
cerr << "Error opening file for reading!" << endl;
return ;
}
int r=1;
// Read male block rooms
cout << "\nMale Block:" << endl;
for (int i = 0; i < mindex; i++) {
inFileRooms.read((char*)&lads,sizeof(newstud));
```

```
if(lads.blocktype=='M'){
if(lads.roomnum!=0){
cout << "Room " << lads.roomnum;
cout << "
Student:"<< lads.name << ",
" << lads.id << ", Gender: " << lads.gender << endl;
}
}
}
// Read female block rooms
cout << "Female Block:" << endl;
for (int i = 0; i < findex; i++) {
inFileRooms.read((char*)&lads,sizeof(newstud));
if(lads.blocktype=='F'){
if(lads.roomnum!=0){
cout << "Room " << lads.roomnum;
cout << "
Student:"<< lads.name << ",
" << lads.id << ", Gender: " << lads.gender << endl;
}
}
}
inFileRooms.close();
cout<<"\n\tclick any key to continue\n";
getch();
system("cls");
}
void summary(){
newstud students;
int x=0;
ifstream countdata("newstuds.txt");
if(countdata.fail()){
cerr<<"error opening\n";
exit(1);
}
while(countdata.read((char*)&students, sizeof(newstud)))
x++;
countdata.close();
cout<<"\n\n\tthere are total of "<<x<<" students\n";
int m=0,f=0;
ifstream indata("newstuds.txt");
while(indata.read((char*)&students, sizeof(newstud))){
if(strcmp(students.gender, "Male")==0){
m++;
}
else if(strcmp(students.gender, "Female")==0){
f++;
}
}
indata.close();
cout<<"\n\tthere are "<<m<<" male students and "<<f<<" female students\n";
cout<<"\n\tclick enter to continue\n";
getch();
system("cls");
```

```
}
void about(){
cout << "
DEPARTMENT OF COMPUTER SCIENCE " << endl << endl;
cout << "
COLLEGUE OF COMPUTING " << endl << endl;
cout << "
GROUP :- 4" << endl << endl;
cout << "
--" << endl;
cout << "
| No. |
Name
|
Id
|" << endl;
cout << "
|
--" << endl;
cout << "
| 1
|
|
|" << endl;
cout << "
| 2
|
|
|" << endl;
cout << "
| 3
|
|
|" << endl;
cout << "
| 4
|
|
|" << endl;
cout << "
| 5
|
|
|" << endl;
cout << "
| 6
|
|
|" << endl;
cout << "
---" << endl << endl;
cout << "
Press any key to continue !!" << endl;
getch();
```

```
cout<<"\n\n\tassigned succesfully\n";
break;
case 6:
assignByCgpa();
cout<<"\n\n\tassigned succesfully\n";
break;
case 7:
displayDorms();
break;
case 8:
deleteStudent();
break;
case 9:
summary();
break;
case 10:
about();
break;
default:
system("cls");
cout << "
---" << endl;
cout << "
THANK YOU
" << endl;
cout << "
FOR ACCESSING IOT DORMITORIES
" << endl;
cout << "
---" << endl;
cout << "
-" << endl << "
-" << endl << "
-" <<
endl << "
-" << endl;
getch();
cout << "
BYE";
choice=11;
}
} while (choice != 11);
}
```

CONCLUSION:

In conclusion, this C++ program serves as a comprehensive and effective tool for managing student assignments within university dormitories. By offering functionalities to accept, assign, delete, find, and update student records, it ensures efficient and accurate handling of data. The program's robust design caters to the dynamic needs of dormitory administration, streamlining the process and minimizing potential errors. Through this application, universities can maintain organized and up-to-date records, ultimately enhancing the overall dormitory management experience.