**Session 5 (unit-3): Classes and Object and concept of Encapsulation**

1. Create a class Student having fields name, rollno, division, id and marks in 3 subjects. The class also has three methods accept\_details(),  average\_marks() and display\_details() as member procedures. Accept the details of two students as command line arguments and Display their information.

#include<iostream>

using namespace std;

class Student

{

private:

char name[50];

int rollno;

char div[5];

int id;

int marks1;

int marks2;

int marks3;

int avg;

public:

void accept\_details();

void average\_marks();

void display\_details() ;

};

void Student::accept\_details()

{

cout << "Student Details:" << endl;

cout << "Student name: ";

cin >> name;

cout << "Student Roll no: ";

cin >> rollno;

cout << "Student Id: ";

cin >> id;

cout << "Student Division: ";

cin >> div;

}

void Student::average\_marks()

{

cout << "\nPlease enter Student marks1 :" << endl;

cin>>marks1;

cout << "\nPlease enter Student marks2 :" << endl;

cin>>marks2;

cout << "\nPlease enter Student marks3 :" << endl;

cin>>marks3;

avg=(marks1+marks2+marks3)/3;

cout << "\n Student avgerage marks:" <<avg<< endl;

}

void Student:: display\_details()

{

cout<< "\nStudent Details are:" << endl;

cout<< "\nStudent Roll no is: " << rollno;

cout<< "\nName of the Student is: " << name;

cout<< "\nDiv of student is: " << div;

cout<< "\nId of the Student is: " << id<<endl;

}

int main(int argc, char \*argv[])

{

if (argc>1)

{

cout<<"Too many arguments supplied. \n"<< argv[1];

}

else

{

cout<<"The argument supplied is clear and One argument expected.\n";

}

Student b1, b2;

b1.accept\_details();

b1.display\_details();

b1.average\_marks();

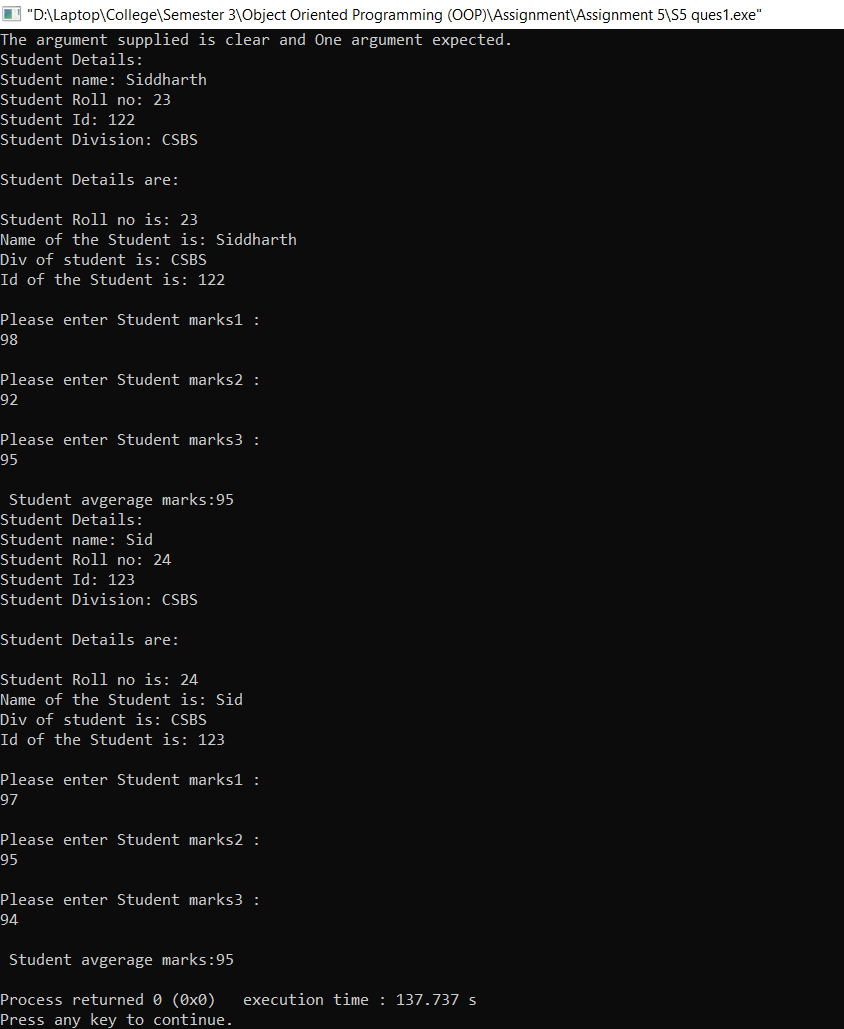
b2.accept\_details();

b2.display\_details();

b2.average\_marks();

return 0;

}



**2.Write a program by creating an 'Employee' class having the following methods and print the final salary.**

* 1. **'getInfo()' which takes the employee Id, his salary and number of hours of work per day of employee as parameter**
  2. **'AddSal()' which adds $10 to salary of the employee if it is less than $500.**
  3. **'AddWork()' which adds $5 to salary of employee if the number of hours of work per day is more than 6 hours.**

#include<iostream>

using namespace std;

class employee

{

private:

char employee[50];

int ep\_salary;

int ep\_id;

int ep\_work;

int ep\_new1\_salary;

int ep\_new2\_salary;

int ep\_tot\_salary;

public:

void getinfo();

void AddSal();

void AddWork() ;

};

void employee::getinfo()

{

cout << "employee Details:" << endl;

cout << "employee name: ";

cin >> employee;

cout << "employee Id: ";

cin >> ep\_id;

cout << "employee hour of work : ";

cin >> ep\_work;

cout << "employee salary:";

cin >> ep\_salary;

cout<<"\n";

cout<< "\nEmployee Details are:" << endl;

cout<< "\nName of the employee is: " << employee;

cout<< "\nId of the employee is: " << ep\_id<<endl;

cout<< "\nNumber of hour-work employee is:" << ep\_work;

cout<< "\nEmployee salary is:"<< ep\_salary<<endl;

}

void employee::AddSal()

{

if(ep\_salary<500)

{

ep\_new1\_salary=ep\_salary+10;

cout<<"The salary after incresing 10$ is="<<ep\_new1\_salary;

cout<<"\n";

}

else

{

ep\_new1\_salary=ep\_salary;

cout<<"Your salary is expected salary\n"<<endl;

}

}

void employee:: AddWork()

{

if(ep\_work>6)

{

ep\_new2\_salary=5+ep\_new1\_salary;

cout<<"\nThe salary after incresing 5$ is="<<ep\_new2\_salary<<endl;

}

else if(ep\_new1\_salary>500)

{

ep\_tot\_salary = (ep\_new1\_salary + ep\_new2\_salary) - ep\_salary;

cout << "The total extra salary is=" << ep\_tot\_salary << endl;

}

else if (ep\_new2\_salary > 500)

{

ep\_tot\_salary = (ep\_new1\_salary + ep\_new2\_salary) - ep\_salary;

cout << "The total extra salary is=" << ep\_tot\_salary << endl;

}

else

{

cout << "You want more bonus and salary go get to the work and do some hardwork on your work\n"<< endl;

}

}

int main()

{

employee b1, b2;

b1.getinfo();

b1.AddSal();

b1.AddWork();

b2.getinfo();

b2.AddSal();

b2.AddWork();

return 0;

}

