**Question 1: (30 points)**

Assume that we have the following classes:

|  |  |
| --- | --- |
| class CourseGrade  {  public:  CourseGrade(int cid, float ch, int sid, int grade);  ~CourseGrade();  float GetCH();  int GetSID();  int GetCID();  int GetGrade();  void SetCH(float ch);  void SetSID(int sid);  void SetCID(int cid);  void SetGrade(int g);  private:  int cid; // course id  int sid; // student id  int grade; //the grade the student of id **sid** obtained in the course of id **cid**  int ch; //credit hours of the course out of 100  }; | class Student  { public:  Student(int id, string name);  Student();  ~Student();  void ComputeGPA();  void AddCourseGrade(CourseGrade);  virtual bool IsInGoodStatus()=0;  virtual bool IsMaster() = 0;  float GetGPA();  void SetName(string);  string GetName();  void SetID(int);  int GetID();  protected:  CourseGrade \* courses;  int count,size;  private:  string name;  int id;  float gpa; }; |

The CourseGrade class is used to represent a grade of a student. Assume that the class is implemented and ready for use.

For the Student class, each student has id, name, and gpa. The **gpa** is calculated based on the **courses** taken by the student using ComputeGPA member function. The **courses** member variable tracks the courses taken by a student using a dynamic array of CourseGrade objects. The **count** represents the number of these courses, while **size** represents the array size which could be different than count in certain times.

1. (6 points) Write the code of **Student(int id, string name)** constructor according to the above description.

When a new student object is created, the student has no courses in the beginning.

1. (12 points) Write the code for **void AddCourseGrade(CourseGrade CG)** function which receives a CourseGrade Object named CG and adds it to the list of courses taken by the student, make sure the function applies the following:

* If the received CourseGrade CG is already in the list of taken **courses** for that student, a grade update should happen with the grade value in CG.
* If the received CG object is the first course to be added, the function should create a dynamic array of size 5 using **courses** data member, and adds the course to the first location. The size will be 5 and the count will be 1 in this case.
* If the array of **courses** is already created and count <size, the CG object will be added to the array.
* If the array of **courses** is already created and count =size, then a new array will be created by a new size equals to the old size +5, the old array objects will be copied to the new one, and the new CG object will be added to the new array and the old array will be deleted.

1. (12 points) Write the code for **void ComputeGPA()** member function that computes the **gpa** data member of the student. Note, the gpa is computed by the summation of each course grade multiplied by its credit hours, then divided by total credit hours of these courses.

**Question 2: (20 points)**

Now assume that we want to expand **Question 1** to cover two types of students: Bachelor and Master students. A master student has a master thesis while a bachelor student has a graduation project. The passing grade system for master students is different from the one for bachelor students. For a master student, the gpa should be >=75 to be in good status, while for bachelor it should be >=60. The passing grade for a master student for an individual course is 70 while it is 50 for a bachelor student.

Define a new class called MasterStudent that is extended from Student class. In this new class, define the following

1. (3 points) Data member variable thesis of type string to store thesis title.
2. (3 points) One Constructor **MasterStudent(int id, string name, string thesis)**
3. (3 points) A setter and getter for the thesis member variable.
4. (5 points) A member function that overrides the **IsMasterStudent()** function inherited from Student class, that returns true.
5. (6 points) A member function that overrides the **IsInGoodStatus()** function inherited from Student class. A master student is considered in a good status if and only if he passes all the courses he has taken with a grade >=70, and has a gpa of 75 or higher.

**Question 3: (30 points)**

Continuing on questions 1 and 2. Assume that MasterStudent class and BachelorStudent class are fully implemented and ready to use with all needed constructors, setters, getters, and member functions mentioned before. Complete the following:

1. (8 points) Write a non-member function **Student \* ReadOneStudent()** that asks the user to enter the type of the student (1 for Master and 2 for Bachelor) and the details of the student to create a dynamic object of the selected type, inputs its data and return a pointer to it as a result.
2. Write a main function that does the following:
   1. (3 points) Define an array of Student pointers of size 100.
   2. (5 points) Read 100 students information and storing them into the array defined above, use ReadOneStudent() function to simplify your job.
   3. (8 points) Assume we have an array of type CourseGradeGrade of size 2000, assume that it is already loaded with data, loop through the array, for each CourseGrade, add the object to the corresponding Student using AddCourse member function defined before. Note, each student should have his/her courses only, use student ID to verify that identity of the student.
   4. (6 points) Print the total number of master students who are not in good status, also print their names.

**Question 4: (20 points)**

Trace the following program and write the generated output in the box below only?

|  |  |
| --- | --- |
| # include<iostream>  using namespace std;  class A{  public: A(int[]);  A();  A operator+(A);  int operator+(int);  int operator\*(A);  void print();  private:  int data[3];  };  A::A(){}  A::A(int values[]) {  for (int i = 0; i<3; i++)  data[i] = values[i];  }  void A::print() {  for (int i = 0; i < 3; i++)  cout << data[i] << " ";  cout << endl;  }  int A::operator\*(A a) {  int res = 0;  for (int i = 0; i<3; i++)  res += data[i] \* a.data[i];  return res;  }  A A::operator+(A a) {  A temp;  for (int i = 0; i<3; i++)  temp.data[i] = data[i] + a.data[i] + 2;  return temp;  }  int A::operator+(int k) {  int m = k;  for (int i = 0; i<3; i++)  k += data[i];  return m;  }  int main() {  int d1[] = { 1, 2, 3 };  int d2[] = { 1, 3, 4 };  A a3, a1(d1), a2(d2);  int uu, uu2;  a1.print();  a3 = a1 + a2;  a3.print();  uu = a1\*a2;  uu2 = a1 + 15;  cout << uu << endl << uu2 << endl; return 0;  } | class A {  public:  static int c;  int id;  A()  {  id = c++; cout << "AC" << id << endl;  }  ~A()  {  cout << "AD" << id << endl;  }  };  class B : public A {  public:  B() { cout << "BC" << id << endl; }  ~B() { cout << "BD" << id << endl; }  };  int A::c = 0;  int main()  {  A a1;  B \*pb;  pb = new B[3];  B b1;  return 0;  }  [Grab your reader’s attention with a great quote from the document or use this space to emphasize a key point. To place this text box anywhere on the page, just drag it.]  [Grab your reader’s attention with a great quote from the document or use this space to emphasize a key point. To place this text box anywhere on the page, just drag it. |
|  |  |