**Final Exam, Spring 2014/2015**

**Question 1: ( 20 points)**

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

|  |  |
| --- | --- |
| class A{  public:  A(int,int);  A operator+(A);  A operator+(int);  int operator\*(A);  void print();  private:  int x, y;  };  A::A(int v1, int v2)  { x = v1; y = v2; }  void A::print()  { cout << x << " " << y << endl;}  int A::operator\*(A a)  { x = x\*a.x;  y = y\*a.y;  return x + y;  }  A A::operator+(A a)  { x = x+a.x;  y = y+a.y;  A b(x,y);  b.x = b.x + 3;  b.y = b.y + 4;  return b;  }  int main()  {  A a1(3,2), a2(4,2);  int uu,uu2;  a1.print();  a1 = a1 + a2;  a1.print();  uu = a1\*a2;  a1.print();  uu2 = uu++;  cout << uu << 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 b1;  B \*pb;  pb = new B;  return 0;  } |
|  |  |

**Question 2: ( 24 points)**

Each student has a Name, ID, and GPA. Write a C++ class to track information about students. Name your class Student. Within the class, define the following:

1. Member variables to track Name, ID, and GPA. ( 3 points)
2. Setters and Getters for each member variable with appropriate validation for the GPA and ID. GPA should be ≤ 100 and ≥ 0 , while ID should be >0. ( 6 points)
3. A member function called GetRating which returns a string. The function returns the status according to the following table: ( 6 points)

|  |  |
| --- | --- |
| **GPA** | **Rating** |
| ≥ 84 | Excellent |
| ≥ 76 and <84 | Very Good |
| ≥ 68 and <76 | Good |
| ≥ 60 and <68 | Fair |
| Otherwise | Fail |

1. Write a main function that:
   * Asks the user to input the number of the students in the class.
   * Then define an array of students for the given number.
   * Finally, read the students information (ID, Name, GPA) and store in the defined array, and finally print a report of the students’ ratings. (9 points)

**Question 3: ( 16 points)**

Assume we have 3 x 3 Matrix. Different operations can be performed against the matrix as shown in below example.

|  |  |
| --- | --- |
|  | Note: the first value 35 represents multiplying first row from A with first column from B: 2\*5+3\*3+4\*4=10+9+16=35 |
| Note: the first value 7 represents adding first cell from A with first cell from B: 2+5=7 | Note: the first value 6 represents multiplying 3 with first cell from A: 2\*3=6 |

Complete below class and define additional functions as the following:

class Matrix {

public:

Matrix( ) {

for (int r = 0; r < 3; r++)

for (int c = 0; c < 3; c++)

values[r][c] = 0;

}

private:

int values[3][3];

};

1. Overload + operator to allow adding two matrices, the function should return a new matrix.
2. Overload \* operator to allow multiplying two matrices as explained above, the function should return a new matrix.
3. Overload \* operator to allow multiplying a matrix with a constant, the function should return a new matrix.
4. Overload stream output operator << to allow printing a matrix object, print the matrix object row by row.

**Question 4: ( 20 points)**

A business hires two types of employees: Full Time and Part Time. Full time employees are paid their salary based on number of **years of experience (100**  JD per year up to 10 years) and **rank** (three ranks 1: 500 JD additional, rank 2: 300 JD additional, and rank 3: 100 additional). Part time employees are paid their salary based on **hourly rate** and **number of hours** worked per month. All employees have a base salary. Given the following **Employee** class:

class Employee {

public:

Employee( string n):name(n) { }

void SetName(string n) { name=n;}

string GetName() { return name; }

float GetBaseSalary(){ return 250;}

virtual float GetSalary(){ return 0;}

private:

string name;

};

Write a new class for full time employees. Name your class as FTE. FTE class should track employee’s name, years of experience and rank. For FTE class, define the following (8 points):

1. A default constructor ,
2. SetValues that sets the name, years of experience, and rank and
3. GetSalary which computes the salary as given above and adds to it the base salary.

Assume that you have finished the FTE class correctly, and assume that a class for part time employees (PTE) is already implemented correctly with a default constructor, SetValues(hourly rate, Hours), and GetSalary. Write a main function that does the following (12 points):

1. Define an Array size 20 to track the employees of both types.
2. Read the information of 20 employees and store them inside the array defined in the previous step. The user will be asked to specify the type of employee first, and then input data according to the type of employee (full time or part time).
3. Once all data has been entered, compute the average salary of all entered employees using the array of information loaded in the previous step.

**Question 5: ( 20 points)**

Assume we have the following two classes implemented and ready for use:

|  |  |
| --- | --- |
| class WheelsSet  {  public :  void Move(float distance\_in\_meter);  // negative means backward  void Turn(float angle);  // negative means left  ...  }; | Class Battery  {  public:  void Discharge(float percentage);  void Charge(float percentage);  float GetLevel();  ...  }; |

Each battery has a specific energy level, the Discharge function in battery class consumes (reduces) the energy level of the battery by the given percentage, while Charge function increases the energy level by the given percentage.

Each Robot has a Wheel set and a Battery. Write a class that represents the Robot. The robot should be able to provide the following functions:

1. int MoveForward(float meters) : this allows the robot to move forward by the given distance, to do that it needs to move the wheelsSet forward. Note, each 10 meters consumes 1% of the battery level. Also, a robot cannot move if it has no enough battery for the move operation.
2. int MoveBackward(float meters) : this allows the robot to move backward by the given distance, to do that it needs to move the wheelsSet backward. Note, each 10 meters of move consumes 1% of the battery level
3. void TurnLeft(float angle): this allows the robot to turn to the left by the given angle, to do that it needs to turn the wheelsSet. Note, each 360 degrees of move consumes 1% of the battery level
4. void ChargeFull(): this allows the robot to charge its battery level to 100%

**Question 1) ( 16 points)**

1. When class B is inherited from class A, what is the order in which the destructor of those classes are executed

|  |  |
| --- | --- |
| 1. Class A first Class B next 2. Class B first Class A next | 1. Class B's only as it is the child class 2. Class A's only as it is the parent class |

1. If a class Demo1 has a pure virtual function named f1, and class Demo2 inherits from Demo1, then, which of the following is correct
2. It is optional for Class Demo2 to define the implementation of function f1
3. A user can define an object instance from class Demo1
4. It is optional for Class Demo1 to define the implementation of function f1
5. Class Demo2 cannot be inherited.
6. One of the following is correct
7. Static member functions can access class member variables but non-static cannot
8. You can call a static member function using object instance or using class name
9. Non-static functions cannot access static class variables
10. You can call a non-static member function using object instance or using class name
11. The default access level assigned to members of a class is \_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| 1. Private 2. Public | 1. Protected 2. Needs to be assigned |

1. For which type of class private and protected members of the class can be accessed from outside the same class in which they are declared

|  |  |
| --- | --- |
| 1. No such class exist 2. Friend | 1. Static 2. Virtual |

1. Having a member function defined as int f1(int x, int y) const means
2. The function f1 cannot change x or y
3. The function f1 cannot return a value that is based on calculations of class member variables
4. The function f1 cannot change the class member variables
5. The function f1 cannot change any value iniside, it only returns
6. The advantage of defining a member function as int f1(const A & x)
7. Guarantees that the passed object x is not changed within the function
8. Allows quick parameter passing by passing the object value by reference
9. No advantages
10. A and B
11. Inside a function definition for a member function of an object with data element x, which of the following is equivalent to this->x:

|  |  |
| --- | --- |
| 1. (\*this).x 2. \*this->x | 1. x 2. Both A and C are correct |

**Question 2**) What is the output of the following **(20 points)**:

|  |  |
| --- | --- |
| class test {  public:  test();  test operator- (int m);  int n;  static int xyz;  };  test::test() { n = 17; xyz++; };  test test::operator- (int m)  { n -= (m - 3); return \*this; };  int test::xyz = 9;  int main() {  cout << test::xyz << endl;  test b[5];  cout << b[2].n << endl;  b[2] = b[3] - 2;  cout << b[2].n << endl;  cout << b[3].n << endl;  cout << b[3].xyz << endl;  return 0;  } | class A {  public:  A(int a = 4);  void SetX(int a);  int GetX();  virtual void print();  private:  int x;  };  A::A(int a ) { x = a; }  void A::SetX(int a) { x = a; }  int A::GetX() { return x; }  void A::print() { cout << x << endl; }  class B : public A {  public:  B(int h = 3);    virtual void print();  private:  int y;  };  B::B(int h ){  y = h; A::SetX(h + 13);  }  void B::print() { cout << (\*this).y + this->GetX() << endl; }  int main(){  B b1, b2(2), \*b3;  A \*data[4];  data[0] = new B(4);  data[1] = new A(5);  b1.print();  b2.print();  b3 = new B(3);  data[2] = b3;  data[0]->print();  data[1]->print();  data[2]->print();  return 0;  } |
|  |  |

**Question 3**) **(16 points)**

Each city has a name, number of residents (مقيمين أو سكان ), and average yearly income (دخل) per resident.

**class City {**

**public:**

**City (string name, int numResident, float avgIncome);**

**City ();**

**float GetCityIncome();**

**private:**

**string name;**

**float income;**

**int numResidents;};**

Write the implementation of the following functions:

1. **GetCityIncome** function. This computes the total yearly income of the city according to the given income and number of residents. ( 4 points)
2. **Overload the + operator** such that it allows combing two cities into one new city object. The combination should reflect the merging of two cities including the name, income and number of residents. Note that the average income of the combination should reflect correctly for the total of the two merged cities. The name should include both names. (7 points)
3. **Overload the > operation** to compare between the two city objects and return true if the left city has more income. ( 5 points)

**Question 4**) **(24 points)**

A company has a computer system that has five subsystems. The company has three types of employees Accountant (محاسب), HR (شؤون موظفين), and Admin (ادارة). Each employee has permissions (صلاحيات) to access one or more of the five subsystems based on his/her employment type. The company offers salaries based on the employee type and rank with base salary computed based on rank. Given the following two classes:

|  |  |
| --- | --- |
| class Permissions  {  public:  Permissions();  Permissions(bool sys1,bool sys2,bool sys3,bool sys4, bool sys5);  ~Permissions();  void SetPermissions(bool sys1, bool sys2, bool sys3, bool sys4, bool sys5);  bool HasAccess(int system\_number);    private:  bool sys1, sys2, sys3, sys4, sys5;  }; | class Employee {  public:  Employee(int rank, string name);  virtual Permissions GetPermission() = 0;  double GetBaseSalary();  virtual double GetSalary()=0;  string GetName();  int GetRank();  protected:  int rank;  string name;  }; |

Assume that all member functions (non-virtual) defined in both classes are implemented.

Define a new class called Accountant. This class should

1. Inherit (derive) from Employee class and include one new int member variable ( years of experience). (6 points)
2. Define a constructor to initialize all member variables (rank, name, years of experience) (6 points)
3. Implement the **GetPermissions** function such that the Accountants have access to the first two subsystems only (sys1, and sys2). (6 points)
4. Implement the **GetSalary** function such that for each rank, the accountant receives 150JD in addition to the base salary that is given to all employees. The ranks will be 1,2,3, or 4. (6 points)

**Question 5**) **(24 points)**

Assume that the classes for Permissions and Employee are all implemented correctly. Also, assume that the classes for Accountant, Admin, and HR are implemented correctly with an initializer constructor for each class receives (rank, name), and also each class has implemented GetSalary and GetPermissions functions.

1) Write a non-member function **Employee \* ReadEmployee()** which takes no parameters and returns a pointer to an Employee object. The function asks the user to enter employee information as the following: **rank, name, type** where rank could be 1, 2, 3, or 4 and type is 1 for Accountant, 2 for HR, and 3 for Admin. Your function should create and set an object of the appropriate type according to the user input and return a pointer to it. (6 points)

2) Write a main function that:

1. Defines an array of employee **pointers** with size 1000. (4 points)
2. Read 1000 employee objects using ReadEmployee function done before and store them in the array. (3 points)
3. Print the salaries of the 1000 employees. (3 points)
4. Read 100 requests of employees requesting to get permissions to access one of the subsystems, out of these 100 requests, print the total number of valid request (valid request means the employee has access to that requested subsystem).

Requests comes in the following format: Employee name, subsystem number

ex. Ahmad, 2 🡪 means employee Ahmad is requesting to access subsystem 2.   
Assume that employees’ names are unique (no repetition in names) (8 points)

**Question 1: ( 20 points)**

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

|  |  |
| --- | --- |
| class A{  public: A(int, int);  A operator+(A);  A operator+(int);  void operator++();  int operator\*(A);  void print();  private:  int x, y;  };  A::A(int v1, int v2) {  x = v1; y = v2;  }  void A::print() {  cout << x << " " << y << endl; }  int A::operator\*(A a)  { x = x\*a.x\*2;  y = y\*a.y;  return x + y; }  void A::operator++()  { x = x+2;  y = y+3; }  A A::operator+(A a)  { x = x + a.x;  y = y + a.y;  A b(x, y);  b.x = b.x + 5;  b.y = b.y + 6;  return b; }  int main()  { A a1(3, 2), a2(2, 1);  int uu, uu2;  a1.print();  a1 = a1 + a2;  a1.print();  uu = a1\*a2;  a2++;  a1.print();  a2.print();  uu2 = uu++;  cout << uu <<" "<< uu2 << endl;  return 0; } | class A {  public:  A() { x = 3; }  A(int a) :x(a){}  void SetX(int a) { x = a; }  int GetX() { return x; }  virtual void print()  { cout << x << endl; }  private:  int x;  };  class B : public A {  public:  B(int h) :y(h) { A::SetX(3); }  B() { y = 5; }  void print()  { cout << (\*this).y + this->GetX() << endl; }  private:  int y;  };  int main()  {  B b1, b2(2), \*b3;  A \*a1 = new B(4);  A \*a2 = new A(3);  b1.print();  b2.print();  b3 = new B(5);  b3->print();  a1->print();  a2->print();  return 0;  } |
|  |  |

**Question 2: ( 24 points)**

**Part A:** Write a C++ class named Patient. Separate the class definition from implementation. For the Patient class, define the following member variables of access type private: (**4 points for class definition**)

1. id of type int
2. name of type string
3. age of type int
4. numOfTemp of type int
5. temperatures of type int \*

and define the following methods (member functions/constructors) of access type public: ( **1 point each**)

1. SetID to set the id
2. SetName to set the name value
3. SetAge to set the Age value
4. GetID to get the value of id
5. GetName to get the name value
6. GetAge to get the age value
7. SetTempratures ( will receive an array of temperatures and it is size)
8. one constructor ( id, name, age,numOfTemp )
9. Another constructor with no parameters ( will set numofTemp to 3).
10. Destructor

Note, constructors will dynamically create an array for temperatures with size determined by numOfTemp values.

**Part B: (10 points)**

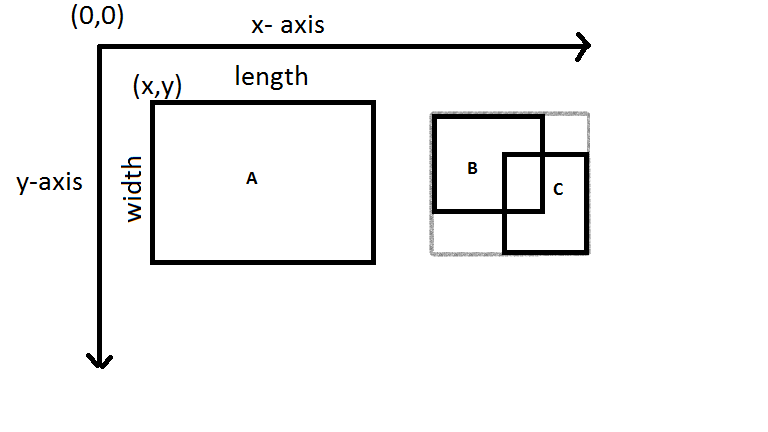
Write the main function that does the following:

1. Define 2 arrays of temperatures T1 and T2. For array T1 use { 20, 30, 31,32} , For array T2 use { 31, 31,32}
2. Define two patient objects, name them **p1 , p2**.
3. Set array T1 to p1 and T2 to p2.
4. Set the id, name, and age of p1 to 10, “Ahmad”, and 32.
5. define one array of patients named **patients** of size 4.

**Solution of question 2**

**Question 3: (16 points)**

Assume we have a Rectangle class. Each rectangle has a left upper corner with (x,y) coordinates and has a length and width as shown in diagram below where three rectangles A, B, and C are shown. The x-axis increases from left to right, and the y-axis increases from top to bottom.



**Complete the class below and define additional functions as the following:**

class Rectangle {

public:

Rectangle(int x, int y, int Len, int Width);

Rectangle operator+(Rectangle);

bool operator>(Rectangle);

void operator\*(float scale);

void operator++();

private:

int x, y;

int length, width;

};

1. void operator\*(float scale) will scale the current rectangle up (if scale >1) or down (scale <1) by multiplying the width and height with the scale value.
2. void operator++() will shift the current rectangle one point forward on the x-axis direction.
3. bool operator>(Rectangle) will return true if the current rectangle’s area is bigger than the received one. Hint: the area of a rectangle is calculated by multiplying the width by the length
4. Rectangle operator+(Rectangle) will return a new rectangle, the new rectangle should be the smallest that can contain both rectangles (current and received). For example, B+C in the above figure will return a new rectangle that contains both B and C

**Solution of question 3**

**Question 4: ( 20 points)**

Each building is located at a specific city and has a total area size measured in square meters. A building could be a commercial one (بناية تجارية), or a residential one(بناية سكنية). For commercial buildings, the tax is computed based on the area size, along with how much occupied size (المساحة المستخدمة). For residential buildings, the tax (الضريبة) is computed based on area size and type, where type is ‘A’, ‘B’, or ‘C’.

Assume that we already have below Building class implemented and ready for use.

class Building

{

public:

Building(string city, float area\_size);

Building();

~Building();

void SetAreaSize(float);

void SetCity(string);

string GetCity();

float GetAreaSize();

virtual float ComputeTax() = 0;

protected:

string city;

float area\_size;

};

Build a new class called CommercialBuilding that extends (derives from) Building class and includes the following:

* Add a new member variable called occupied\_area\_size of type float. (**5 points**)
* Define a constructor that receives city,area\_size and a new member variable called occupied\_area\_size of type float. (**5 points**)
* Override ComputeTax function that such it computes the tax in JD according to the following formula: (**10 points**)

If the city is Amman, then

Tax= (30\*occupied\_area\_size +10\* (area\_size - occupied\_area\_size)

For all other cities

Tax= (20\*occupied\_area\_size +5\* (area\_size - occupied\_area\_size)

**Solution of question 4**

**Question 5: ( 20 points)**

In addition to Building class, Assume that the following CommercialBuilding and ResidentialBuilding classes are implemented and ready for use:

|  |
| --- |
| class CommercialBuilding: public Building  {  public:  CommercialBuilding(string city, float area\_size, float occupied\_area\_size);  ~CommercialBuilding();  float ComputeTax();  private:  float occupied\_area\_size; };  class ResidentialBuilding : public Building  {  public:  ResidentialBuilding(string city, float area\_size, char type);  ~ResidentialBuilding();  float ComputeTax();  private:  char type; }; |

1. Define a non-member function called **Building\* ReadBuilding()** that asks the user to enter the building choice (R for residential or C for commercial) and then reads city, area\_size, and type or occupied\_area\_size and then create and return a pointer to an object of the appropriate building type.
2. Write a main function that includes:
   1. Define an array Building \* buildings[100];
   2. Read 100 buildings information using ReadBuilding function and store them inside buildings array.
   3. Print the total tax due for buildings in Amman.