SECTION A: STRUCTURED QUESTIONS [70 MARKS]

Question 1 [8 marks]

Consider the following **Program 1(a).** Answer questions (i), (ii) and (iii) below.

```
// Program 1(a)
#include <iostream>
using namespace std;

class Distance {
   private:
      int feet,inches;
   public:
      Distance(int f=0, int i=0){
         feet = f;
         inches = i;
       }

      Distance operator()(int scale) {
           feet   = feet * scale;
           inches = inches * scale;
           return *this;
      }

      int operator[](int index) {
           if (index==0) return feet;
           if (index==1) return inches;
           else return 0;
      }

      void display()const{
         cout <<feet <<" feet and "<<inches <<" inches" <<
         endl;
       }
};

int main() {
   Distance p(10, 2), q, r(9,88);

   cout << "Output 1 : ";
   p.display();

   q = p(5);
   cout << "Output 2 : ";
   q.display();

   cout << "Output 3 : ";
   p.display();

   cout << "Output 4 : " << "Value = " << r[0] << endl;
   cout << "Output 5 : " << "Value = " << r[1] << endl;
   cout << "Output 6 : " << "Value = " << r[2] << endl;
   return 0;
}
```

(i) Trace the output of **`Program 1(a).`** (3 marks)

(ii) Suppose the program is going to be added with another class named `Metric` and a regular function named `distanceInMeters` as given in the following program segment. However, the new version of the program is unable to compile due to the compiling error **"feet and inches of class `Distance` are private"**. Add the appropriate code to the class `Distance` in order to fix this error. Write your answer in the space provided in **`Program 1(b).`** (2 marks)

```
class Metric{
     private:
          double km;

     public:
          Metric(const Distance &d)
          {  // Get distance in feet
             double totalFeet = d.feet + d.inches / 12.0;

            // Convert to metric unit, i.e. kilometers
                km = totalFeet * 0.0003048;

          }
};

double distanceInMeters(const Distance &d)
{   // Get distance in inches
    double totalInches = d.feet*12 + d.inches;

    // Convert to meters
    double m = totalInches * 0.0254;

    return m;
}

```

(iii) Add a copy constructor to the class `Distance` which rounds up the copied distance to the nearest feet. For example, if the source distance is 2 feet 8 inches, it will be copied as 3 feet, and if it 2 feet 3 inches, it will be 2 feet. Note that 1 foot is equivalent to 12 inches. Write your answer in the space provided in **`Program 1(b).`** (3 marks)

```
// Program 1(b)

#include <iostream>
using namespace std;

class Distance {
   private:
      int feet,inches;
   public:
       Distance(int f=0, int i=0){
          feet = f;
          inches = i;
        }

      Distance operator()(int scale) {
           feet   = feet * scale;
           inches = inches * scale;
           return *this;
      }

      int operator[](int index) {
           if (index==0) return feet;
           if (index==1) return inches;
           else return 0;
      }


      void display()const{
        cout <<feet <<" feet and "<<inches <<" inches" <<
        endl;
       }


      // Answer space for the question (ii)













```

```
        // Answer space for the question (iii)













};
```

Question 2 **[17 marks]**

Consider the following **`Program 2`**. The class **`NumDays`** is meant to store the number of **`hours`** worked and convert it to a number of **`days`**. For example, 15 hours will be converted to 1.875 days and 25 hours to 3.125 days. The overloaded + operator will be used when two **`NumDays`** objects are added together. It returns an object of class **`NumDays`** that summing up the **`hours`** with the passed object's **`hours`** member. The overloaded – operator will be used when one **`NumDays`** object is subtracted from another. It returns an object of class **`NumDays`** that calculates the difference of the **`hours`** with the passed object's **`hours`** member. The overloaded ++ (prefix) operator should increment the number of hours stored in the object. When incremented, the number of **`days`** should be automatically re-calculated. The overloaded -- (postfix) operator should decrement the number of **`hours`** stored in the object. When decremented, the number of days should be automatically re-calculated.

Complete **`Program 2`** in (i), (ii), (iii) and (iv) to define each of the overloaded operators accordingly.

```
// Program 2

#include <iostream>
using namespace std;

class NumDays
{  private: double hours, days;

   public:
      NumDays(double h = 0)
         { hours = h; days = h / 8; }
      void setHours(double h)
         { hours = h; days = h / 8; }
      double getHours() const
         { return hours; }
      void setDays(double d)
         { days = d; hours = d * 8; }
      double getDays() const
         { return days; }

    //  (i)     overloaded + operator                    (4.5 marks)










    //  (ii)     overloaded – operator                   (4.5 marks)










    //  (iii)     overloaded prefix ++  operator         (3.5 marks)










    // (iv)    overloaded postfix -- operator                (4.5 marks)










}; // end of class NumDays

int main()
{
   NumDays one(25), two(15), three, four;

   cout << " One: " << one.getDays() << endl;
   cout << " Two: " << two.getDays() << endl;
   three = one + two;
   cout << " Three: " << three.getDays() << endl;
   four = ++three;        // Prefix increment
   cout << " Four = ++Three: " << four.getDays() << endl;
   four = three--;        // Postfix decrement
   cout << " Four = Three--: " << four.getDays() << endl;
   return 0;
}
```

The screen output of the program is as shown below:

```
        One: 3.125
        Two: 1.875
        Three: 5
        Four = ++Three: 5.125
        Four = Three--: 5.125
```

Question 3 **[20 marks]**

Consider the following **`Program 3`** and answer questions (i), (ii) and (iii).

```
// Program 3

#include <iostream>
using namespace std;

class Officer {
private :
      string name;
      int id;
public :
    Officer (string oname = "", int oid = 0) {
        name = oname; id = oid;}

    void set(string a, int b) {
        name = a; id = b;  }

    void print() {
        cout << "Officer   : " << name << " " << id << endl; }
};

class Offender {
private :
      string name, ic, address;
public :
    Offender (string oname = "", string id = "",string add =
    ""){
        name = oname; address = add; ic = id; }

    void set(string a, string b, string c) {
        name = a;  ic = b;  address = c;}

    void print() {
        cout << "Offender :" << name << " " << ic << " "
            << address << endl; }
};

class Summon {
private :
     int code;
     Officer *officer;
     Offender offender;
public :
    Summon (int cod = 0, string name ="", string id = "",
            string add = "" ){
       code = cod; offender.set(name,id,add); }

    void addOfficer(string a, int b) {
       officer = new Officer;  officer->set(a,b); }

    void removeOfficer() {
       delete officer;  officer = NULL; }
52
53      void print() {
54          cout<<endl<<"Summon Code : "<<code<<endl;
55          offender.print();
56          officer->print();
57         }
58  };
59
60  int main()
61  {
62    Officer  a("ALI", 75381);
63    Offender b("KASSIM","841203015311",
64                "NO 1 JALAN PS 13 TAMAN PERMATA SINAR");
65
66    Summon c(12,"SELAMAT","992112035621",
67              "NO 3 JALAN PC 31 TAMAN PERMATA CAHAYA");
68
69    c.addOfficer ("AHMAD", 87611);
70    a.print();
71    b.print();
72    c.print();
73    c.removeOfficer();
74
75    return 0;
76  }
```

(i) Describe the two approaches to implement the concept of class aggregrations and identify the line number of code in **`Program 3`** which implement these approaches.

(6 marks)

Name of the aggregation approach	**Description of the approach**	**The line number that implements the aggregration approach**

(ii) Based on **`Program 3`**, draw the UML diagram showing all the classes and the relationship between them. (9 marks)

(iii) What is the output of **`Program 3`** ? (5 marks)

Question 4 **[25 marks]**

Given **`Program 4`** as shown below. Answer questions (i), (ii) and (iii).

```
// Program 4

#include <iostream>
using namespace std;

class Parent
{
  private:
    int pnum1;

  protected:
    int pnum2;

  public:
    int pnum3;

    Parent(int a = 10, int b = 20, int c = 40) {
      pnum1 = a; pnum2 = b; pnum3 = c;
      cout << "Parent with pnum1 = " << pnum1 << ", pnum2 = "
           << pnum2 << ", pnum3 = " << pnum3 << endl;
    }

    void print() {
       cout << "Print from Parent class" << endl;
    }

    ~Parent() {
      cout << "Parent object destroyed with pnum1 = "
           << pnum1 << ", pnum2 = " << pnum2 << ", pnum3 = "
           << pnum3 << endl;
    }
}; //End class Parent

class Child1: private Parent
{
  private:
    int cnum1;

  public:
    Child1(): Parent() {
      cnum1 = 50;
      cout << "Child1 with cnum1 = " << cnum1 << endl;
    }

    void print() {
      cout << "Print from Child1 class" << endl;
    }
48
49
50
51      ~Child1() {
52       cout << "Child1 object destroyed with cnum1 = "
53             << cnum1 << endl;
54      }
55  }; //End of class Child1
56
57  class Child2: protected Parent
58  {
59    private:
60      int cnum2;
61
62    public:
63      Child2(int a = 0, int b = 0, int c = 0, int d = 0):
64      Parent(a, b, c) {
65        cnum2 = d;
67        cout << "Child2 with cnum2 = " << cnum2 << endl;
68      }
69
70      void print() {
71          Parent::print();
72          cout << "Print from Child2 class" << endl;
73      }
74
75      ~Child2() {
76        cout << "Child2 object destroyed with cnum2 = "
77             << cnum2 << endl;
78      }
79  }; //End of class Child2
80
81  int main()
82  {
83      Child1 c1;
84      Parent p1(2, 4);
85      Child2 c2(3, 6, 9);
86      c1.print();
87      c2.print();
88      return 0;
89  }
```

(i) What will the lines 83 to 88 print onto the screen when the **Program 4** runs? State your answer by completing the following table: (13 marks)

Line	Output
83	

84	
85	
86	
87	
88	

(ii) Based on the objects created in the **`main`** function of the **`Program 4`** (lines 83 to 85), determine the member variables that each object owns. Write (**Yes**) in the corresponding blank cells, if the object own or has a copy of the variable; and write (**No**) if otherwise. State your answer by completing the following table:

(5.5 marks)

	Member Variables				
Object	**`pnum1`**	**`pnum2`**	**`pnum3`**	**`cnum1`**	**`cnum2`**
`c1`	■				■
`p1`		■			
`c2`			■		

(iii) With reference to the same member variables in (ii), determine whether they are accessible by the object listed below. Write (**Yes**) in the corresponding blank cells, if it is accessible; and write (**No**) if otherwise. State your answer by completing the following table:

(6.5 marks)

	Member Variables				
Object	**`pnum1`**	**`pnum2`**	**`pnum3`**	**`cnum1`**	**`cnum2`**
`c1`					■
`p1`					
`c2`				■	

SECTION B: PROGRAMMING QUESTION TOTAL – 30 MARKS

This section consists of **ONE (1)** question only.

Question

A faculty in a private university wishes to computerise its student records system. The record of each student includes his or her name and the program he or she enrolls. The faculty offers several programs at undergraduate and postgraduate levels such as "Bachelor of Computer Science", "Bachelor of Social Study", "Master of Business Administration", "PhD of Social Science", and many more. Regardless of the program or degree, each student is appointed with a lecturer as his or her academic advisor. The role of the advisor is to guide the student on academic matters such as the rules and regulations of the academic system.

There are two types of students, undergraduate and postgraduate students, respectively. The computer system has to record the current CGPA for each of the undergraduate students. As for the postgraduate programs either master or PhD degrees, they are conducted in fully research-based. Each postgraduate student has to have a research project and a lecturer to supervise his or her project.

Based on the given problem, answer the following questions:

(a) Draw the UML class diagram for the above problem. Your design has to include the classes and their attributes and methods accordingly, as well as relationships between the classes. Each class has to provide a constructor, mutators, and assessors.

(8 marks)

(b) Then, write the C++ code to implement the design. Your implementation should apply object-oriented programming concepts including data hiding, composition or aggregation, and inheritance.

(12 marks)

(c) Next, utilize the classes to store a list of undergraduate students and a list of postgraduate students. You need to use dynamic arrays for the lists and fill them in with data read from two input files accordingly. Figure 1 and Figure 2 show the examples of input files containing the list of undergraduate and postgraduate students, respectively. The first line in each file indicates the number of students the file contains. Following that is the record of a student in which each attribute is arranged in a line. The student records are separated by blank lines. Finally, print all the students from the arrays into another text file. Figure 3 shows an example of the output file.

(10 marks)

```
100                             } The number of students

Ali Bin Bakar                   } Student's name and program
Bachelor of Science             }
Prof. Dr. Kamil Ahmad           } Academic advisor
3.85                            } Student's current CGPA

Aminah Binti Ahmad
Bachelor of Business
Dr. Rohaya Qamarool
3.99

 (due to limited spaces, the remaining student records are not shown in this figure)
```

Figure 1: An example of input file named "ugstudents.txt", for the list of undergraduate student records. Note that, the texts in *italic* are given to describe the fields accordingly. Also, only two records from the input file out of 100 are shown here due to limited spaces.

```
50                                  } The number of students

Bakar Bin Malik                     } Student's name and program
Master of Social Science            }
Dr. Rohaya Qamarool                 } Academic advisor
Prof. Dr. Kamil Ahmad               } Project supervisor
The Effects of Social Media         } Research project

Ruby Gabriella
PhD of Business
Prof. Dr. Kamil Ahmad
Prof. Dr. Saidi Abdullah
Adaptive Human Capital Management

(due to limited spaces, the remaining student records are not shown in this figure)
```

Figure 2: An example of input file named "pgstudents.txt", for the list of postgraduate student records. Note that, the texts in *italic* are given to describe the fields accordingly. Also, only two records from the input file out of 50 are shown here due to limited spaces.

```
THE LIST OF UNDEGRADUATE STUDENTS
No   Name                    Program                  Advisor                   CGPA

  1  Ali Bin Bakar           Bachelor of Science      Prof. Dr. Kamil Ahmad      3.85
  2  Aminah Binti Ahmad      Bachelor of Business     Dr. Rohaya Qamarool        3.99
...  ...............         .....................     .......................   ....
                              (students no 3 to 98 go here)
...  ...............         .....................     .......................   ....

99  Nuraini Binti Zulkifli   Bachelor of Science      Dr. Rohaya Qamarool        3.70
100 Iskandar Binti Muhammad  Bachelor of Electrical   Prof. Dr. Sharom Mat       3.88




THE LIST OF POSTGRADUATE STUDENTS
No Name                    Supervisor               Project

 1  Bakar Bin Malik         Prof. Dr. Kamil Ahmad   The Effects of Social Media
 2  Ruby Gabriella          Dr. Saidi Abdullah      An Adaptive Human Capital Management
..  ...............         .....................   ...................................
                             (students no 3 to 49 go here)
..  ...............         .....................   ...................................
50  Zulkarnain Bin Hashim   Prof. Dr. Kamil Ahmad   Artificial intelligence for robots
```

Figure 3: An example of the output file containing both the lists of undergraduate and postgraduate students. Note that only some students are shown here due to limited spaces.

Answer spaces for Section B

Answer spaces for Section B

Answer spaces for Section B