Question 1  ........................................................................................................... (16 marks)

Given the following code snippet, answer the questions that follow.

```cpp
class Address
{
    public:
        Address(int houseNumber, char *streetName) {
            mHouseNumber = houseNumber;
            mStreetName = streetName;
        }
        int houseNumber() {
            return mHouseNumber;
        }
        char* streetName() {
            return mStreetName;
        }
    private:
        int mHouseNumber;
        char *mStreetName;
};

class Student
{
    public:
        Student(int studentNumber, Address *address) {
            mStudentNumber = mStudentNumber;
            mAddress = mAddress;
        }
        ~Student() {
            delete mAddress;
        }
        void print() {
            cout << "Student number: " << mStudentNumber << endl;
            cout << "Student address: " << mAddress->houseNumber() << " "
                 << mAddress->streetName() << endl;
        }
    private:
```

Name and surname: ____________________________
Student number: ____________________________

Question 1  ........................................................................................................... (16 marks)

Given the following code snippet, answer the questions that follow.

```cpp
class Address
{
    public:
        Address(int houseNumber, char *streetName) {
            mHouseNumber = houseNumber;
            mStreetName = streetName;
        }
        int houseNumber() {
            return mHouseNumber;
        }
        char* streetName() {
            return mStreetName;
        }
    private:
        int mHouseNumber;
        char *mStreetName;
};

class Student
{
    public:
        Student(int studentNumber, Address *address) {
            mStudentNumber = mStudentNumber;
            mAddress = mAddress;
        }
        ~Student() {
            delete mAddress;
        }
        void print() {
            cout << "Student number: " << mStudentNumber << endl;
            cout << "Student address: " << mAddress->houseNumber() << " 
                 " << mAddress->streetName() << endl;
        }
    private:
```
```cpp
int mStudentNumber;
Address *mAddress;
}

int main()
{
    int houseNumber;
    char *streetName = new char[1024];
    cout << "Please provide the student's address" << endl;
    cout << "House number: \"; cin >> houseNumber;
    cout << "Street name: \"; cin >> streetName;
    cout << endl;
    Address address(houseNumber, streetName);
    Student student(12345678, &address);
    student.print();

    return 0;
}
```

After compiling and running the program, you get a segmentation fault. In order to detect the problems, you decide to use third-party tools.

1.1 Which tool discussed in class can help you identify segmentation faults? (1)

1.2 Which flag should be added to your gcc compilation command to add debugging information so that the third-party tools will work. (1)

1.3 After running the tool from (1.1), the following is printed after doing a backtrace: 0x0000000000400da5 in Student::print (this=0x7fffffffde00) at student.cpp. Rewrite two lines of the code from the program above to fix the segmentation fault for the given error. (2)

1.4 After fixing the error above, you get another segmentation fault: double free or corruption (out): 0x00007fffffdff0, 0x0000000000400d08 in Student::Student (this=0x7fffffffde00, inchrg=<optimized out>) at student.cpp. Give two alternate solutions to fix the fault. (2)

1.5 Now that the segmentation faults are gone, there is still a memory leak that needs fixing. Describe what a memory leak is. (2)
1.6 Which tool discussed in class can help you identify memory leaks? (1)

1.7 Besides memory leaks, name two other memory errors that can be detected using the tool in (1.6). (2)

1.8 Which of the following flags should be passed to the tool from (1.6) to show a full trace instead of a summary: - -leak-check, - -leak-resolution, - -show-reachable, or - -dummy-answer. (1)

1.9 You get the following memory leak: at 0x4C2B800: operator new[](unsigned long) (in vgpreload_memcheck-amd64-linux.so) by 0x400AFF: main (main.cpp:6). Shortly explain two alternate solutions to remove the leak without changing the program intent. (2)

1.10 What is the difference between delete ptr; and delete [] ptr; in C++? (2)

---

**Question 2**

Given an incomplete code snippet from practical five for a 10x10 matrix, identify and describe the seven programming mistakes.

```cpp
int main ()
{
    int a[][10];
    int *b = new int[5];
    int i, r1, c1=10;
    cout << endl << "Enter elements of matrix 1:" << endl;
    for(i=0; i<=r1; ++i)
        for(j=0; j<c1; j++)
            {cout << "Enter element at A["<< i+1 << "]["<< j+1 << "]:";
             cin >> a[i][j];
            }
    delete a;
    delete b;
}```
Question 3  .................................................................................................................. (8 marks)
For each of the following class declarations, determine if the declaration and the inheritance is valid and will compile (true) or if it is invalid and will give a compilation error (false) in C++. The declarations in the table below are independent of each other.

<table>
<thead>
<tr>
<th>Number</th>
<th>Mistake Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Declaration</th>
<th>True/False</th>
</tr>
</thead>
<tbody>
<tr>
<td>class ClassA;</td>
<td></td>
</tr>
<tr>
<td>abstract class ClassA{};</td>
<td></td>
</tr>
<tr>
<td>class ClassA{}; class ClassB{}; class ClassC : ClassA, ClassB{};</td>
<td></td>
</tr>
<tr>
<td>class ClassA{}; class ClassB{}; class ClassC : public ClassA, private ClassB{};</td>
<td></td>
</tr>
<tr>
<td>template&lt;class T&gt; class ClassA{}; class ClassB : protected ClassA&lt;int&gt;{};</td>
<td></td>
</tr>
<tr>
<td>template&lt;class T&gt; class ClassA{}; class ClassB : protected ClassA&lt;ClassB&gt;{};</td>
<td></td>
</tr>
<tr>
<td>template&lt;class T&gt; class ClassA{}; class ClassB : protected ClassA{};</td>
<td></td>
</tr>
<tr>
<td>template&lt;class T&gt; class ClassA{}; template&lt;class S&gt; class ClassB : protected ClassA&lt;S&gt;{};</td>
<td></td>
</tr>
</tbody>
</table>

Question 4  .................................................................................................................. (19 marks)
The following code illustrates the Prototype design pattern. The programmer is very inexperienced and a
lot of mistakes (both programmatic and logical) have crept in. Rewrite the code of both classes and fix all mistakes. You are not allowed to change the intent of the code, the main function, the copy constructors’ parameters, or the classes’ member variable declarations. All member variables must be deep copied and the program must output the following:

```
1 Init A: 0
2 Init B: 1
3 Copied A
4 Copied B
5 Deleted B
6 Deleted A
7 Deleted B
8 Deleted A
```

The code is given as:

```cpp
class Prototype
{
Prototype()
{
  mA = new double[0];
  cout << "Init A: " << mA << endl;
}
Prototype(const Prototype& other){
  mA = other->mA;
  cout << "Copied A" << endl;
}
"Prototype(){
  delete [] mA;
  cout << "Deleted A" << endl;
}
Prototype* clone();

private:
  int *mA; // Not allowed to change.
};

class ConcretePrototype : Prototype
{
ConcretePrototype()
{
  mB = new int[1];
  cout << "Init B: " << mB << endl;
}
ConcretePrototype(const ConcretePrototype& other){
  mB = other->mB;
  cout << "Copied B" << endl;
}
"ConcretePrototype(){
  cout << "Deleted B" << endl;
}
Prototype* clone(){
  return new Prototype(this);
}

private:
  int mB; // Not allowed to change.
};

int main() // Not allowed to change.
```

{
    Prototype *p1 = new ConcretePrototype();
    Prototype *p2 = p1->clone();
    delete p1;
    delete p2;
    return 0;
}