1 Question 1

1.1 Complete Main for Input Age

MainFunction

```
QString userInput = QInputDialog::getText(0,”DOB”,”Enter Date of Birth (Format dd/mm/yyyy)”);
QStringList strDOB = QString.split(“/”);
QList<int> IntDOB;
For (int I = 0; I < strDOB.size(); i++)
{
IntDOB.append(strDOB[i].toInt());
}
Int userAge = computerAge(IntDOB[0]/intDOB[1]/intDOB[2]));

QMessageBox::Information(0,”Your Age”, userAge);
```

1.2 DateList manages a list of QDate. Explain three different ways of designing such a class.

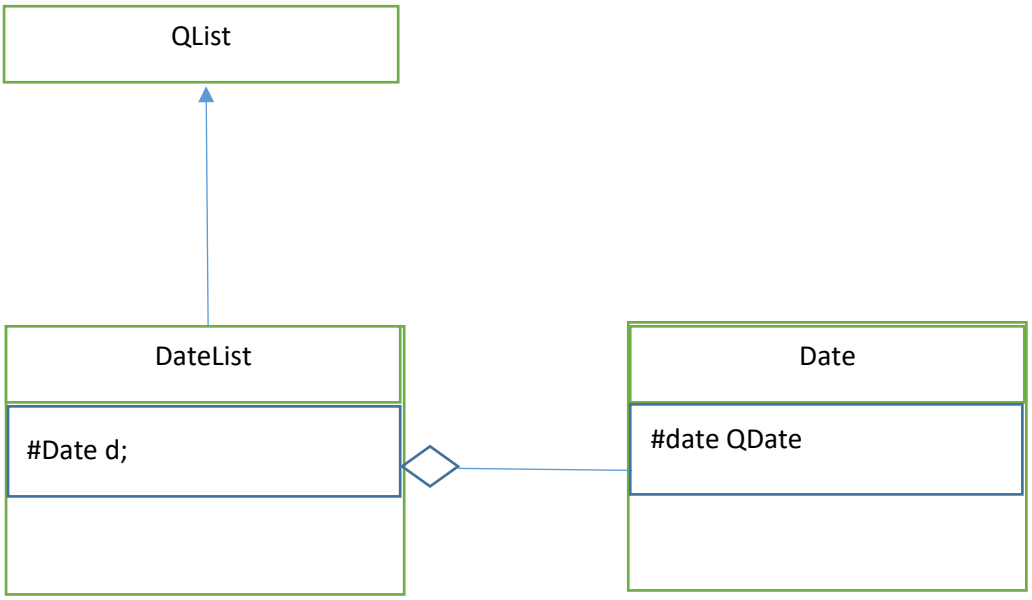
1) class DateList : public QList;

2) You can create a QList<QDates> as a member function.

3) You can have a DateList class which has *QDate objects

```
#include QDate
#include QList
#include
```

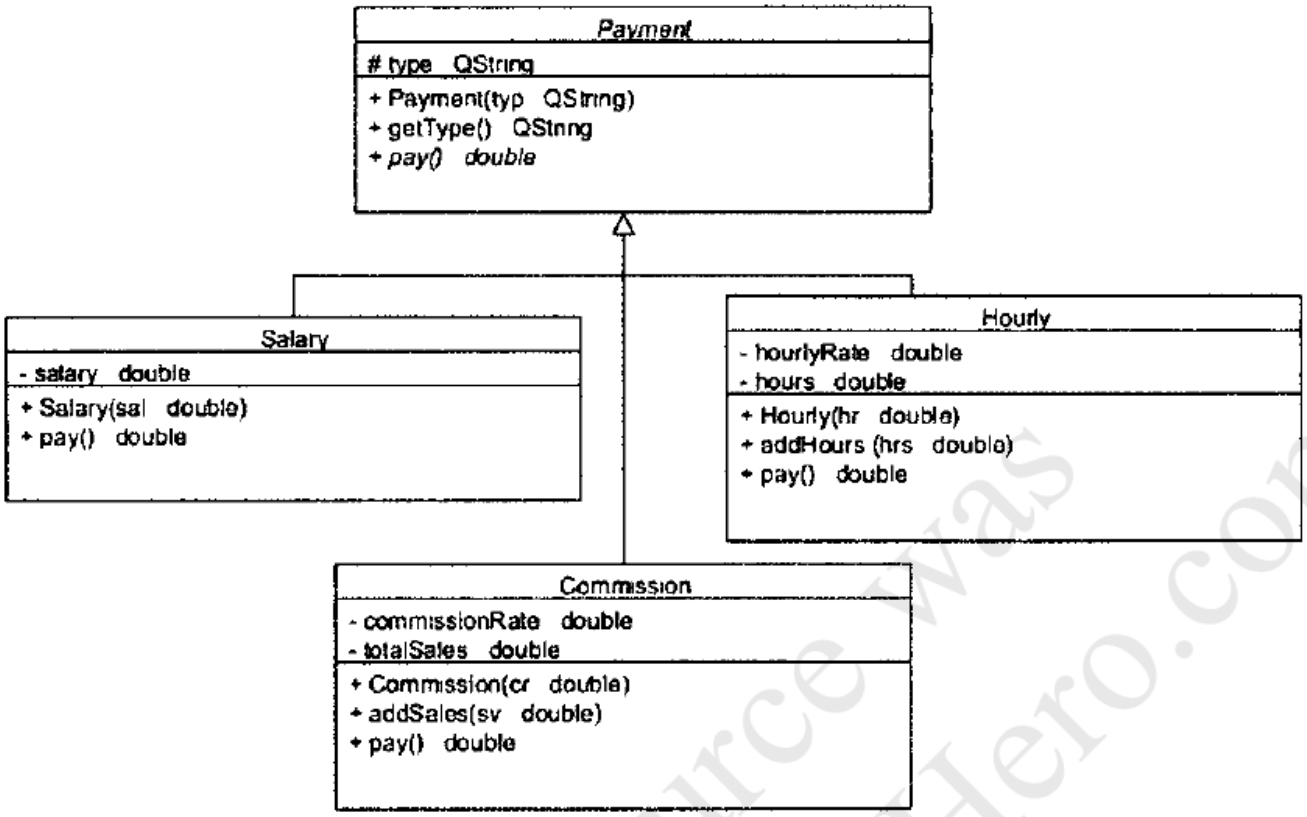
1.3 Draw a UML representation of the relationship between QDate and DateList where QDate instances exist after a DateList Instance has been destroyed.



1.4 Name the widget class in QT framework that is only used to input a date

QCalendarWidget

2 UML Diagram



2.1 Virtual double pay() = 0;

2.2 You cannot instantiate a pure virtual function

2.3 Hourly::Hourly(double hr) : Payment("Hourly"){

HourlyRate = hr;

Hours = 0.0;

}

2.4 Polymorphic assignment is when you assign create an instance of a child class by assigning it to its base class eg. Payment h = new Hourly(400.00);

2.5 Polymorphism is when the compiler at run time decides which function call to use. In this case the which implementation of pay() will be decided at run time.

2.6 virtual toString();

2.6.1 QString Payment::toString() const{
Return QString ("Type: %1).arg(type);

2.6.2 QString Salary::toString() const{
Return QString("%1, HourlyRate: %2, Hours Worked: %3)

.arg(Payment::toString())

.arg(hourlyRate)

.arg(hours);

2.7 What does this do? Void PaymentGroup addPayment(Payment *p) { p->setParent(this);

Sets the argument passed into it as a base class so that you can have a list of objects

2.8 When the parent is deleted, the children will also be deleted.

2.9 Composite Design Pattern

2.10 The concrete classes are the composite and the leaf classes, the abstract class is the Component. The concrete classes inherit from the Component class and the composite class has a component object

2.11 QList<Payment*> PaymentList::getPayments(QString typ)

```
QList <Payment*> ListofTyp;
Foreach (Payment *p, PaymentList)
 {
 If (p->getType() == typ)
 append(p);
 }
 return ListofTyp;

 }
```

2.12 QList<double>;

3 Telephone Gui

3.1 Insert code

```
4    #include <QPushButton>
5    #include <QLineEdit>

6    class NumberPad   public QWidget {
7    /* insert code */ Q_OBJECT
8    public
9        NumberPad(),
10   /* insert code */  public slots
11       void numberClicked(),
12   private
13       /* insert code */  QPushButton *clickedButton;
14       QLineEdit* display,
15   },
16   #endif // NUMBERPAD_H
```

numberpad.cpp

```
17   #include "numberpad h"
18   #include <QGridLayout>

19   NumberPad  NumberPad() {
20       QGridLayout* grid = new QGridLayout(this),
21       QString operatorValues[10] = { "0","1","2","3","4","5","6",
22       "7","8", "9"},
23       /* insert code */  QPushButton *numberButtons[10]
24       grid->addWidget(display, /* insert code */),  0,0
```

```
25      for (int i = 0, i < 10, i++){
26         numberButtons[i] =
27         connect
28         grid->addWidget(numberButtons[i], (int)i/3 + 1, i%3 ),
29      }
30      this-> setLayout(/* insert code */),
31  }

32  void NumberPad  numberClicked (){
33      QPushButton *clickedButton =
34      QString numberValue =
35      display->setText(display->text()
36  }
```

New QPushButton(operatorValues[i])

(numberButton[i], SIGNAL(clicked()),this, SLOT(numberClicked());

Qobject-cast<QPushbutton*> (sender());

clickedButton->text();

+numberValue);

DON'T GET THIS

3.2

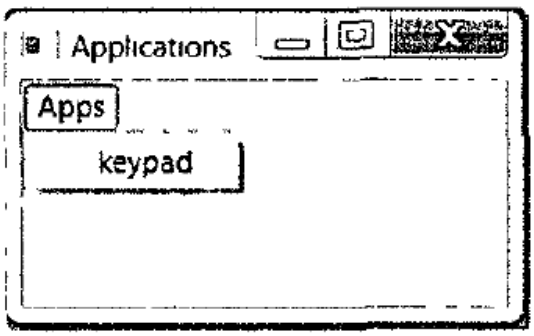


Figure 1

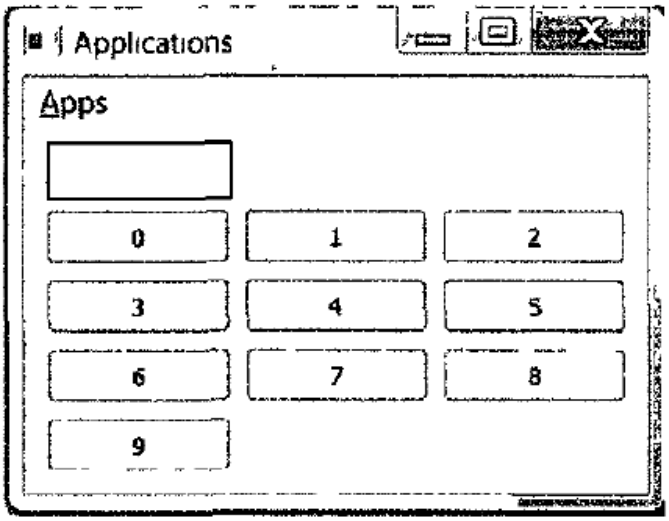


Figure 2

3.2.1 Write Class Head

Class MyMainWIndow : public QMainWindow{

3.2.2 Class MyMainWIndow : public QMainWindow{
QObject

QToolBar *mBar = New QToolBar();

QMenu *apps = new QMenu("Apps");

QAction *keypad = new QAction("keypad");

Apps->addAction(keypad);

Public slots

Connect(keypad,SIGNAL(triggered(),this, SLOT(showNumberpaid());

3.2.3 MyMainWIndow::showNumberPad()

{

numberPad n;

n.show()

}