University of Leeds

School of Computing

PRF – Programming Fundamentals
January 2006 – Marking Scheme

Question 1

(a)

The original program is:

```
#!/usr/bin/python

# numbers.py
#
# Converts a number to words.
#

words = ['Zero', 'One', 'Two', 'Three', 'Four', 'Five', 'Six', 'Seven', \
         'Eight', 'Nine']

number = raw_input ("Enter Number: ")

# Now Display "number" as Words
# So "458" is displayed as "Four Five Eight", and so on.
```

The solution is is to add a simple loop, viz:

```
#!/usr/bin/python

# numbers.py
#
# Converts a number to words.
#

words = ['Zero', 'One', 'Two', 'Three', 'Four', 'Five', 'Six', 'Seven', \
         'Eight', 'Nine']

number = raw_input ("Enter Number: ")

for d in number:
  print words [int (d)],

print
```

(b)

The trick here is to use the `string` module with its handy constants.

```
#!/usr/bin/python

# stringDigits.py
#
# Checks that a String contains only digits.
#

import string

def onlyDigits (s):
  for c in s:
    if c not in string.digits:
      return False

  return True

s = "123456"
if onlyDigits (s):
  print s, "contains only digits."
else:
  print s, "contains something that is not a digit!"

s = "12XZ3456"
if onlyDigits (s):
  print s, "contains only digits."
else:
  print s, "contains something that is not a digit!"
```

(c)

```
#!/usr/bin/python

# numbers.py
#
# Converts a number to words.
#

import string

def onlyDigits (s):
  for c in s:
    if c not in string.digits:
      return False

  return True
```

```
#
# Main Program
#

words = ['Zero', 'One', 'Two', 'Three', 'Four', 'Five', 'Six', 'Seven', \
         'Eight', 'Nine']

number = raw_input ("Enter Number: ")

if not onlyDigits (number):
  print 'That string contains something that is not a digit!'
else:
  for d in number:
    print words [int (d)],

  print
```

(d)

```
#!/usr/bin/python

# numbers.py
#
# Converts a number to words.
#

import string
import sys

def onlyDigits (s):
  for c in s:
    if c not in string.digits:
      return False

  return True

#
# Main Program
#

if len (sys.argv) != 2:
  print 'Usage:', sys.argv[0], '<number>'
elif not onlyDigits (sys.argv[1]):
  print 'That string contains something that is not a digit!'
else:
  words = ['Zero', 'One', 'Two', 'Three', 'Four', 'Five', 'Six', 'Seven', \
           'Eight', 'Nine']

  for d in sys.argv[1]:
```

```
    print words [int (d)],

  print
```

In each case above, **[5]** marks for spot on, simple solution, using the given code and with neat output. 1 mark off for each deviation from this. Deviations include arcane variable names, poor commenting, errors handling command-line arguments, etc.

Question 2

The original `Person` class is:

```
public class Person {

  public String name;
  public int age;

  public Person ()
  {
    name = "";
    age = 0;
  }

  public Person (String name, int age)
  {
    this.name = name;
    this.age = age;
  }

  public String getName ()
  {
    return name;
  }

  public void setName (String name)
  {
    this.name = name;
  }

  public int getAge ()
  {
    return age;
  }

  public void setAge (int age)
  {
    this.age = age;
```

```
  }
}
```

(a)

The two obvious errors are: the instance variables are all declared as `public`, and the mutator for age does not validate the value passed to it. **[2]** for spotting each one, and **[2]** for repairing each. Arguably the parameterised constructor also needs a test for the validity of the age, and the name should be validated (it should not be zero length) in the two places it is set. Marks are also available for these if the explanation is sound.

(b)

[6] for spot on commenting, leaving nothing out. **[4]** if some holes could be picked, for example over-commenting or errors in descriptions. **[2]** if commenting is generally acceptable but has some serious flaws.

(c)

A `toString` method is required here **[2]**. Up to **[4]** marks for the implementation – full marks requires handling the grammar if the person is one year old.

(d)

[2] for building class as a sub-class.
[2] for modifying the `Person` class to make instance variables protected.
[4] for the constructors.
[4] for validating the garage
[4] for validating the route number.
[2] for a comprehensive main method.
[2] for general neatness, commenting, etc.

```
/**
 * BusDriver.java
 *
 * Class to store details of Bus Drivers.
 *
 * @author AMJ
 * @version 1.0
 */

public class BusDriver extends Person {

  /*
   * Instance Variables.
   */
  private String garage;
  private String route;
```

```
/**
 * Constructor.
 *
 * All attirubutes are initialised. Name and age are passed to the
 * super-class. Garage and Route are set to empty strings.
 */
public BusDriver ()
{
  super ();
  garage = "";
  route = "";
}

/**
 * Constructor.
 *
 * All attributes are initialised to the values supplied as parameters.
 * They are not validated.
 *
 * @param name Initial name.
 * @param age Initial age.
 * @param garage Initial garage.
 * @param route Initial route.
 */
public BusDriver (String name, int age, String garage, String route)
{
  super (name, age);
  this.garage = garage;
  this.route = route;
}

/**
 * Accessor for Garage.
 *
 * @return Current garage of the Bus Driver.
 */
public String getGarage ()
{
  return garage;
}

/**
 * Mutator for Garage.
 *
 * Garage is set to the supplied value, provided that the value is
 * one of those allowed.
 *
 * @param garage New garage for the Bus Driver.
 * @return True if the garage is changed, false otherwise.
 */
public boolean setGarage (String garage)
{
```

```
  if (garage.equals ("Hunslet") || garage.equals ("Roseville Road")
      || garage.equals ("Bramley") || garage.equals ("Kirkstall")) {
    this.garage = garage;
    return true;
  }
  else {
    return false;
  }
}

/**
 * Accessor for Route.
 *
 * @return Current route of the Bus Driver.
 */
public String getRoute ()
{
  return route;
}

/**
 * Mutator for Route.
 *
 * Route is set to the value supplied, provided it is no longer than
 * three characters and includes only 'X', 'C', 'A' and digits.
 *
 * @param route New route for the Bus Driver.
 * @return True if the value is set, false otherwise.
 */
public boolean setRoute (String route)
{
  if (route.length () > 3) {
    return false;
  }

  for (int i = 0; i < route.length (); i ++) {
    if (route.charAt (i) != 'X' && route.charAt (i) != 'A' &&
          route.charAt (i) != 'C' &&
          !Character.isDigit (route.charAt (i))) {
      return false;
    }
  }

  this.route = route;
  return true;
}

/**
 * String representation.
 *
 * The format is "<name> is <age> years old.".
 *
```

```
 * @return String representation of the Bus Driver.
 */
public String toString ()
{
  String s = name;

  if (age == 1) {
    s += " is 1 year old and is a Bus Driver.\n";
  }
  else {
    s += " is " + age + " years old and is a Bus Driver.\n";
  }

  s+= "The garage is " + garage + " and the route is " + route + ".";

  return s;
}

/**
 * Main
 */
public static void main (String args[])
{

  System.out.println ("Creating Bill ...");
  BusDriver bill = new BusDriver ("Bill", 14, "Bramley", "72");
  System.out.println (bill);

  System.out.println ("Changing Bill to Bob ...");
  if (bill.setName ("Bob")) {
    System.out.println (bill);
  }
  else {
    System.out.println ("Not Allowed!");
  }

  System.out.println ("Setting Garage to \"Leeds\" ...");
  if (bill.setGarage ("Leeds")) {
    System.out.println (bill);
  }
  else {
    System.out.println ("Not Allowed!");
  }

  System.out.println ("Setting Garage to \"Kirkstall\" ...");
  if (bill.setGarage ("Kirkstall")) {
    System.out.println (bill);
  }
  else {
    System.out.println ("Not Allowed!");
  }
```

```
    System.out.println ("Setting Route to \"63B\" ...");
    if (bill.setRoute ("63B")) {
      System.out.println (bill);
    }
    else {
      System.out.println ("Not Allowed!");
    }

    System.out.println ("Setting Route to \"63A\" ...");
    if (bill.setRoute ("63A")) {
      System.out.println (bill);
    }
    else {
      System.out.println ("Not Allowed!");
    }

    System.out.println ("Setting Route to \"X84\" ...");
    if (bill.setRoute ("X84")) {
      System.out.println (bill);
    }
    else {
      System.out.println ("Not Allowed!");
    }

    System.out.println ("Setting Route to \"63\" ...");
    if (bill.setRoute ("63")) {
      System.out.println (bill);
    }
    else {
      System.out.println ("Not Allowed!");
    }
  }
}
```

AMJ, 9th November 2005