RMIT UNIVERSITY

SCHOOL OF SCIENCE & TECHNOLOGY

EEET2482 – SOFTWARE ENGINEERING DESIGN

TUTORIAL WEEKS 9/10

Task 1: In this task, we extend the employee management system that were developed in weeks 7 and 8. In particular, we will allow different salary calculation for different types of Employee as described below.

1. Salary for a customer Service personnel is calculated as below.
   salary = (standard working hour * standard hourly rate of pay * casual rate) + (overtime working hour * 2 * standard hourly rate of pay).
   If the employee is a fulltime employee, casual rate is 1
   If the employee is a part-time employee, casual rate is 2
   If the employee is a casual employee, casual rate is 3

2. Salary for an Engineer is calculated as below.
   salary = (standard working hour * engineer hourly rate of pay) + (overtime working hour * 3 * engineer hourly rate of pay).
   If the engineer has a diploma or a bachelor, engineer hourly rate of pay = standard hourly rate of pay
   If the engineer has a master, engineer hourly rate of pay = standard hourly rate of pay *1.5
   If the engineer has a master, engineer hourly rate of pay = standard hourly rate of pay *2

Update class Employee, class Engineer and class CustomerSerevice according to reflect these updates. You are expected to implement polymorphism to allow runtime checking and invoking of correct functions for specific Employee objects.

In the driver code, create an array of Employee to test the functionalities of the classes.

Task 2: Write a C++ program which takes in integer values from the console and stores them in a linked list. The program should use a switch statement to allow the user to select the following options

- Add a node to the linked list/create a linked list with a single node
- Print the contents of the entire linked list
- Reverse the linked list
- Delete the linked list

Each of the above options can be performed in functions which are external from the class.

The class template from which each node (object) will be instantiated from is

```
class node {
	public:
		node(int num) { data = num, next = NULL; }
		~node() { ; }
		int getData() { return data; }
		void setData(int num) { data = num; }
		void setNext(node *ptr) { next = ptr; }
		node *getNext() { return next; }
	private:
		node *next;
		int data;
};
```