1. (10 marks)

a) (5 marks) List the four main sections of a requirements document.

1. System description
2. Glossary
3. Non-functional requirements
4. Functional requirements

b) (5 marks) List the four main stages of the software development process.

1. Requirements gathering
2. (Object-oriented) analysis
3. (Object-oriented) design
4. Implementation

2. (40 marks)

a) (10 marks) Sketch the use case diagram for Example #2.
b) (15 marks) Consider the use case Sell ice cream (Retail) for Example #2 given in the handout. Assuming that ice-cream cones are sold in the majority of cases, rewrite this use-case with basic, alternative, and exception path flows of events.

USE CASE NAME: Sell Ice Cream (Retail)

ACTORS: Customer, Clerk, Credit Card Company

PRECODITIONS: None

FLOW OF EVENTS: BASIC PATH

1. Use case starts with customer selecting retail ice cream order.
2. System requests flavour.
3. Customer supplies flavour.
4. System requests receptacle type (cone or tub).
5. Customer supplies receptacle-type cone.
8. System requests credit card payment.
9. Customer gives credit card number.
10. System contacts credit card company to request payment.
11. Credit card company confirms payment.
12. Customer receives requested ice cream.
13. This use case ends.

FLOW OF EVENTS: ALTERNATIVE PATH

[In 5.] If receptacle type is tub, go to step 8.

FLOW OF EVENTS: EXCEPTION PATH

[In 3.] If flavour is not available, repeat steps 2-3 up to two more times. If flavour still not available, go to step 13.
[In 7.] If cone-size is invalid, repeat steps 6-7 up to two more times. If cone-size still invalid, go to step 13.
[In 11.] If credit card company denies payment, go to step 13.

POSTCONDITIONS: None
c) (15 marks) Write the use-case *Register pickup complaint* in Example #1 with basic and exception path flows of events.

USE CASE NAME: Register Pickup Complaint

ACTORS: Client, Clerk

PRECODITIONS: None

FLOW OF EVENTS: BASIC PATH

1. Use case starts with customer selecting register pickup complaint.
2. Clerk requests customer name and order ID number.
3. Customer supplies name and order ID number.
4. Clerk requests nature of complaint.
5. Customer supplies nature of complaint.
6. If order is current and order priority is two or less, order priority is increased by one.
7. If order is pending, order priority is increased by three.
8. If order is neither current nor pending:
   a. If order priority is less than or equal to three, fire garbage haulers associated with order.
   b. If order priority is three and customer priority is greater than seven, fire pickup planner associated with order.
9. Clerk issues apology to customer.
10. This use case ends.

FLOW OF EVENTS: EXCEPTION PATH

[In 3.] If order ID number is invalid, reduce customer priority by two and go to step 10.

POSTCONDITIONS: None
a) (15 marks) Consider the following UML domain model for Example #1:

Give the locations (via a numbered circle) and briefly describe five additional mistakes in this domain model relative to the given description.

1. Should be aggregation rather than generalization relation
2. Clerk is subclass of employee
3. Lower bound of 2 not specified
4. Missing class fields, e.g., order priority
5. Depot is associated with cur-order
6. No multiplicities specified for this relation
b) **(10 marks)** Sketch a UML domain model for Example #2.

```
ICMaker
   `-- Clerk
     |     |
     v     v
  ICFBatch
      `-- ICOorder
          |      |
          |      v
          v    v
ICRecipe
```

```
KMCEmployee
```

```
ICOrdRet
     `-- RecType
         `-- ConeSize

ICOrdWhl
     `-- NumTub
```

```
ICOrdWhl
     `-- CustWalk
         `-- CustDist
             `-- Name
                 `-- Address
```

```
Ingredient
    `-- OrdIngredient
```

```
Supplier
    `-- Name
        `-- Address
```

```
ChargeAcct
```

```
CustWalk
```

```
Customer
```

```
Name
```

```
Address
```

```
ChargeAcct
```

```
Name
```

```
Address
```

```
OrdIngredient
```

```
1..10
```

```
1
```

```
2..?
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```

```
I
```