```
MenuItem(String details)
{

    StirngToeknizer str=new StringTokenizer(details,"$$$");
    itemname=str.nextToken();
    price=str.nextToken();
    category=str.nextToken();

    ////////////////////////////////////////////////////OR////////////////////////////////////////////////////
    int i=details.indexOf("$$$");
    itemname=str.substring(0, i));
    str=str.substring(i+3,str.length());
    i=str.indexOf("$$$");
    price=str.substring(0, i));
    category=str.substring(i+3,str.length());

}
```

Q1(a)

1. Correct Extraction of One Field → 1 M
2. Correct Extraction of Two Fields → 1.5 M
3. Correct Extraction of All Three Fields → 2 M

Marking Scheme: 0/1/1.5/2 M

Q1(b-1)

```
public void addMenu(MenuItem m)
{
    if(count<10)
    {
        Items[count]=m;
        count++;
    }
}//End of Method
```

Marking Scheme: 0/0.5/1 M

1. Adding Element → ½ M
2. Updating Count → ½ M

Q1(b-2)

```
public void printReport()
{
        for(int i=0;i<MenuItem.length();i++)
        {
                StringBuffer itemnamebuf=new StringBuffer(MenuItem[i].getItemname());
                itemnamebuf.setLength(30);
                StringBuffer pricebuf=new StringBuffer(MenuItem[i].getPrice);
                pricebuf.setLength(20);
                StringBuffer categorybuf=new StringBuffer(MenuItem[i].getCategory);
                categorybuf.setLength(10);
                System.out.println(itemnamebuf+pricebuf+categorybuf);
        }
}
for(int i=0;i<MenuItem.length();i++)
{
        String str=new String();
        str=str.concat(MenuItem.getItemname());
        for(int j=str.length();j<30;j++)
        {
                str=str.concat(" ");
        }
        str=str.concat(MenuItem.getPrice());
        for(int j=str.length();j<50;j++)
        {
        str=str.concat(" ");
        }
        str=str.concat(MenuItem.getCategory());
        for(int j=str.length();j<60;j++)
        {
        str=str.concat(" ");
        }

        System.out.println(str);
        }
}//End of Class
```

Marking Scheme: 0/1/1.5/2 M

```
// Q2 a(1) → 0/1 M  (1/2 * 2 = 1M)
Account(int noRegularSerial,int noPremiumSerial)
{
                                        this.noRegularSerial      =      noRegularSerial;
                                        this.noPremiumSerial      =      noPremiumSerial;
}
// Q2 a(2) → 0/ ½ / 1M
public double monthlyCost()
{
double cost=baseprice+(noRegularSerial*regularSerialPrice)+(noPremiumSerial*premiumSerialPrice) ;
return cost;
}
// Q2 a(3) → 0/1/1.5/2 M
public int compareTo(Account acc)
{
      int result=0;
      double thiscost=0,accost=0;
      thiscost=this.monthlyCost();
      accost=acc.monthlyCost();
      if(thiscost<acccost)                  result=-1;
      else      if(thiscost==accost)        result=0;
      else      result=1;
      return result
}
```

Q2(b)

```
public static void sortAccount(Account acc[])
{
    Arrays.sort(acc);
    for(int i=0;i<acc.length;acc++)
    {
            System.out.println(acc[i]);
    }
}
```

Marking Scheme: 0/ ½ /1 M

Question 3 Solution-cum-Marking Scheme

```
public    String    encrypt()
{
          String    encryptedString = "";

          String    m          = super.getMessage() ;    // OR String m = getMessage()→ 0/1 M For Getting Message

          // Validity of NumericMessage                  // 0/1 M For Checking Validity
          int i = 0;
          for(i = 0;  i< m.length();  i++)
                    if (m.charAt(i) < '0' || m.charAt(i) > '9') return null; // Invalid Message

          // Valid Message Encryption, Encrypt The Message For Even Length Messages   → This Part 0/1/2 M

          for(i = 0; i<m.length()-1; i=i+2)
          {
                              encryptedString = encryptedString + m.charAt(i+1);
                              encryptedString = encryptedString + m.charAt(i);
          }

          // 0/1 M for ODD Length, If Length is ODD then simply appending the Last Character as it is
          if( m.length() % 2 != 0)     encryptedString   =          encryptedString + m.charAt(i);

          return encryptedString;
}// End of Method
```

Getting Message From Super class 0/1 M

Validity of Numric Message 0/1 M

Loop → 0/1 M

Swapping Characters 0/1 M

0/1 M

Summary

1. Getting Message From Super Class → 0/1 M
2. Validity of Numeric Message → 0/1 M
3. Even Encryption For Loop and Swapping → 1+1 = 2M
4. Odd Length Encryption → 0/1 M