Problem 1

What does the following Java code print:

```java
class A {
    public int key;
}

class B extends A {
}

public class Problem1 {
    public static void f(A X) {
        A Y = X;
        Y.key = X.key + 1;
    }

    public static void f(B X) {
        B Y = new B();
        Y.key = X.key + 2;
        X = Y;
    }

    public static void main(String[] args) {
        A P = new A();
        P.key = 3;
        B Q = new B();
        Q.key = 10;
        f(P);
        System.out.println(P.key);
        f(Q);
        System.out.println(Q.key);
        P = Q;
        f(P);
        System.out.println(P.key);
    }
}
```

Answer: 4, 10, 11. Note that:

- If you call `f(A)` with the first version of `f`, the effect is to change the `key` field of `A` because `X` and `Y` are both references to the same object as `A`.
- If you call `f(A)` with the second version of `f`, nothing changes because reassigning `X` does not affect the value of `A`.
- Since overloaded methods are resolved by the declared type of the variable, the two calls to `f(P)` in `main` call the first version of `f` and the call to `f(Q)` calls the second version.
Problem 2

(Multiple choice: One correct answer) An abstract class is

D. A class that contains at least one abstract method.

Problem 3

A. What is the problem with the following recursive method to compute $a^n$, for positive integer $n$?

```java
public static double exponent(double a, int n) {
    return a*exponent(a,n-1);
}
```

Answer: There is no base case. The program will go into an infinite loop with $n$ getting more and more negative.

B. Describe how this should be fixed. Your solution should still use recursion. (The code does not have to work if $n < 0$).

Answer: Change the body to

```java
if (n == 0) return 1.0;
else return a*exponent(a,n-1);
```

Problem 4

Suppose that you have a doubly linked list with headers at both ends composed out of nodes defined with the following data fields:

```java
class MyNode {
    private int value;
    private MyNode next;
    private MyNode prev;
}
```

Write a method within MyNode called replace(M), which does the following. Assume that $L$ is a MyNode within a doubly linked list, but not either header, and that $M$ is another MyNode. Calling $L.replace(M)$ should cause $L$ to be replaced by $M$ in the linked list. You should not change the value field of any node, just the next and prev fields.

Answer:

```java
public void replace(MyNode M) {
    M.next = next;
    M.prev = prev;
    M.next.prev = M;
    M.prev.next = M;
}
```

Problem 5

Suppose that we have defined the following generic class for linked lists:
class MyNode<T> {
    private T value;
    private MyNode<T> next;

    public MyNode(T V) { value = V; } // Constructor
    public SetNext(MyNode<T> N) { next=N; } // Setter
}

Show the code you would use to construct a list whose first element is the list [2, 3] and whose second element is the list [6]. You can use either headed or unheaded lists, whichever you prefer.

Answer:

MyNode<Integer> F2 = new MyNode<Integer>(3);
MyNode<Integer> F1 = new MyNode<Integer>(2);
F1.SetNext(F2);
MyNode<Integer> S1 = new MyNode<Integer>(6);
MyNode<MyNode<Integer>> L2 = new MyNode<MyNode<Integer>>(S1)
MyNode<MyNode<Integer>> L = new MyNode<MyNode<Integer>>(F1)
L.SetNext(L2);

L is the desired list.

Problem 6

Suppose that you have constructed a hash table for biographical information in which the keys are Strings, denoting a person, and the values are Integers, denoting the year of their birth. Your table has size 8. The following operations, and the values of the hash function, are shown below. Draw a diagram of the state of the hash table.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Hash Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>HT.put(&quot;Lincoln&quot;, 1809)</td>
<td>hash = 4</td>
</tr>
<tr>
<td>HT.put(&quot;Mozart&quot;, 1756)</td>
<td>hash = 2</td>
</tr>
<tr>
<td>HT.put(&quot;Dickens&quot;, 1812)</td>
<td>hash = 2</td>
</tr>
<tr>
<td>HT.put(&quot;Obama&quot;, 1961)</td>
<td>hash = 7</td>
</tr>
<tr>
<td>HT.put(&quot;Gaga&quot;, 1986)</td>
<td>hash = 4</td>
</tr>
</tbody>
</table>

Answer: