There are 10 questions and you have 35 minutes to complete the test. Each question has exactly one correct answer.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Good luck
For each of the following items, please enter one or more answer A, B, C or D, on the sheet provided.

1. Which of the following does the use of abstract data types (ADTs) not typically improve?
   (a) re-use of program components
   (b) efficiency of programs
   (c) clarity of programs
   (d) independent development of program components

2. Given an algorithm $A$ suppose that $T_A(n)$ is the worst case complexity of $A$ and $E_A(n)$ is the expected case complexity of $A$. We say that one sorting algorithm, $A$, is better than another sorting algorithm, $B$, if $T_B(n)$ is not $O(T_A(n))$, or if $T_B(n)$ is $O(T_A(n))$ but $E_B(n)$ is not $O(E_A(n))$. Then which of the following statements is true?
   (a) Merge-Sort is better than Insertion-Sort which is better than Quick-Sort.
   (b) Quick-Sort is better than Merge-Sort which is better than Insertion-Sort.
   (c) Insertion-Sort is better than Quick-Sort which is better than Merge-Sort.
   (d) Merge-Sort is better than Quick-Sort which is better than Insertion-Sort.

3. The following code is used to performing a sorting algorithm:

   ```java
   public void A(int[] a, int x, int y){
       if(x<y){
           int z = B(a,x,y);
           A(a, x, z);
           A(a, z+1, y);
       }
   }
   ```

   What would be the most informative names for the methods $A$ and $B$?
   (a) $A$ should be called `merge` and $B$ should be called `partition`.
   (b) $A$ should be called `mergesort` and $B$ should be called `merge`.
   (c) $A$ should be called `quicksort` and $B$ should be called `merge`.
   (d) $A$ should be called `quicksort` and $B$ should be called `partition`.
4. Which of the following definitions most accurately captures the statement: \( f(n) \) is \( O(g(n)) \), but \( g(n) \) is not \( O(f(n)) \).

(a) There is some \( a, b > 0 \) such that for all \( c > a \), \( f(c) < b \cdot g(c) \), but for some \( c > a \), \( g(c) > b \cdot f(c) \).

(b) There is some \( a, b > 0 \) such that for all \( c > a \), \( f(c) > b \cdot g(c) \), and for all \( a, b > 0 \), there is some \( c > a \) such that \( g(c) < b \cdot f(c) \).

(c) For all \( a, b > 0 \) there is some \( c > a \), \( g(c) > b \cdot f(c) \), and there is some \( a, b > 0 \) such that for some \( c > a \), \( f(c) < b \cdot g(c) \).

(d) For all \( a, b > 0 \) there is some \( c > a \), \( g(c) > b \cdot f(c) \), and there is some \( a, b > 0 \) such that for all \( c > a \), \( f(c) < b \cdot g(c) \).

5. Suppose that a class `StackBlock` is implemented as follows:

```java
class StackBlock{
    Object[] items;
    int top;

    public StackBlock(){items = new Object[10];}

    public void push(Object o){
        if(top==items.length){
            Object[] tmp = items;
            items = new Object[tmp.length*2];
            top = 0;
            for(int i=0; i<tmp.length;i++) push(tmp[i]);
        }
        items[top++] = o;
    }

    public boolean isEmpty(){return top == 0;}
    public Object peek(){
        if(isEmpty()) throw new Underflow("empty!");
        return items[top-1];
    }
    public Object pop(){
        if(isEmpty()) throw new Underflow("empty!");
        return items[--top];
    }
}
```

What best describes the complexity of the `push` method:

(a) `push` is \( O(n) \) in the worst case, but \( O(1) \) amortised time.

(b) `push` is \( O(n^2) \) in the worst case, but \( O(1) \) in the amortised case.

(c) `push` is \( O(n^2) \) in the worst case, but \( O(\lg n) \) in the expected case.

(d) `push` is \( O(n) \) is the worst case, but \( O(\lg n) \) in amortised time.
6. Suppose that we would like to provide an iterator method for the StackBlock class in Question 5, by providing the follow code in the StackBlock class:

```java
public Iterator iterator(){return new SBIterator();}

class SBIterator implements Iterator{
    int index = 0;
    public boolean hasNext(){//missing code;}
    public Object next(){return items[index++];}
}
```

What is the correct implementation of the missing code?

(a) return index < top;
(b) return index <= top;
(c) return index < items.length;
(d) return index++ <= items.length;

7. A deque (double-ended queue) is implemented using an array called items and left and right indices called left and right respectively. The deque is cyclic (or “wraps around”) so that all space in the array can be used.

The method `pushLeft` adds an item to the left end of the deque and is implemented as follows:

```java
public void pushLeft(char c) throws Overflow {
    if (!isFull()) {
        if (left == -1) left = items.length-1;
        items[left] = c;
    }
    else throw new Overflow('Pushing to full deque'.)
}
```

Which of the following is a correct implementation of the missing lines:

(a) left = left-1;
   if (left == -1) left = items.length-1;
   items[left] = c;
(b) left = (left-1)%items.length;
   items[left] = c;
(c) left = (left+1)%items.length;
   items[left] = c;
(d) left = left-1;
   if (left==right+1) left = (left-1)%items.length;
   items[left] = c;
8. A (singly) linked implementation of a Queue contains the following instance variables:

- `front` — a reference to the front of the queue, that is, the end with the item that has been in longest
- `back` — a reference to the back of the queue, that is, the end with the item that was added most recently

The queue must operate in constant time.

The `enqueue` method can be implemented as follows:

```java
public void enqueue (Object a) {
    if (!isFull()) {
        if (isEmpty()) {
            front = new LinkChar(a,null);
            back = front;
        }
        else back.successor = new LinkChar(a,null);
    }

    << missing code >>
}
else throw new Overflow("Inserting in full queue.");
```

Which of the following is a correct implementation of the missing code?

(a) `else back.successor = new LinkChar(a,null);`
(b) `else back = new LinkChar(a,back);`
(c) `else {
    back.successor = new LinkChar(a,null);
    back = back.successor;
}
(d) `else {
    back = new LinkChar(a,back);
    back = back.successor;
}`
9. The previous method of a linked Simplist with instance variables before, after and window can be implemented as follows:

```java
public void previous() throws OutOfBounds {
    if (!isBeforeFirst()) {
        // << missing code >>
    }
    else throw new OutOfBounds("Fell off the front.");
}
```

Which of the following is an incorrect implementation of the missing code? (The following picture may help)

(a) Link temp = window;
    window = window.successor;
    temp.successor = before.successor;
    before.successor = temp;

(b) Link temp = window.successor;
    window.successor = before.successor;
    before.successor = window;
    window = temp;

(c) Link temp = before.successor;
    before.successor = window;
    window = window.successor;
    before.successor.successor = temp;

(d) Link temp = before.successor;
    before.successor = window;
    window = window.successor;
    window.successor = temp;
10. Assume that Poodle has been defined as a subclass of Dog, and that the code

```java
Stack<Dog> pound = new Stack<Dog>();
Dog brutus = new Dog();
Poodle lillie = new Poodle();
```

is directly followed by one of the following:

```java
i. pound.push(lillie);
   lillie = pound.pop();

ii. pound.push(lillie);
    brutus = pound.pop();

iii. pound.push(brutus);
    lillie = pound.pop();

iv. pound.push(lillie);
    lillie = (Poodle) pound.pop();
```

Which of the above will cause an error?

(a) (i) and (iii)
(b) (ii) and (iii)
(c) (iii) only
(d) (i), (iii) and (iv)