Problem Set 1 **Due: Feb 26** **cs2me3 Winter 2009**

Instructions

1. You are encouraged to work in groups of two. If you cannot find a partner, you can work alone.
2. Please submit **one** copy of the assignment; if you are working with a partner, both names should appear on the assignment.
3. For **Part A** of the assignment, you must submit an electronic copy of your Java application by email to the TA (by the time of the lecture on the due date of the assignment). Note that you will get a zero if your program does not compile.

Part A

Write a Java application which implements the Gram-Schmidt algorithm (which is algorithm 2.6 in the notes). Your application should take as input a file of vectors. That is, you invoke your application with the command

```
gramschmidt file.txt
```

where the format of `file.txt` is as follows:

```
(0,1,0,0)
(2.37,6.0,-20,1)
(-5,99,11,2)
(1,34,7,0)
```

That is, `file.txt` consists of a list of vectors in $\mathbb{R}^n$ (in the example above $n = 4$). Your program has to establish whether the input is a correctly formated list of vectors (and quit if it is not), but if the vectors are all of the same length, and well formated, then you may assume the pre-condition (that is, that the vectors are linearly independent).

Part B

1. Do problem 2.14 in the notes (the "handshakes" problem).

   Solution: We partition the participants into the set E of even persons and the set O of odd persons. We observe that, during the hand shaking ceremony, the set O cannot change its parity. Indeed, if two odd persons shake hands, O decreases by 2. If two

even persons shake hands, O increases by 2, and, if an even and an odd person shake hands, $|O|$ does not change. Since, initially, $|O| = 0$, the parity of the set is preserved (from *Problem-Solving Strategies* by Arthur Engel, chapter 1 (Invariance), exercise 32, with solution on page 19).

2. Do problem 2.20 in the notes. The termination part is shown in the solutions, so you do not have to do that; show the partial correctness.

   Solution: For partial correctness of the palindromes algorithm we show that if the pre-condition holds, and *if* the algorithm terminates, then the post-condition will hold. So assume the pre-condition, and suppose first that A is *not* a palindrome. Then there exists a smallest i_0 (there exists one, and so by the LNP there exists a smallest one) such that $A[i_0] \neq A[n - i_0 + 1]$, and so, after the first $i_0 - 1$ iteration of the while-loop, we know from the loop invariant that $i = (i_0 - 1) + 1 = i_0$, and so line 4 is executed and the algorithm returns F. Therefore, "A not a palindrome" $\Rightarrow$ "return F."

   Suppose now that A *is* a palindrome. Then line 4 is never executed (as no such i_0 exists), and so after the $k = \lfloor \frac{n}{2} \rfloor$-th iteration of the while-loop, we know from the loop invariant that $i = \lfloor \frac{n}{2} \rfloor + 1$ and so the while-loop is not executed any more, and the algorithm moves on to line 8, and returns T. Therefore, "A is a palindrome" $\Rightarrow$ "return T."

   Therefore, the post-condition, "return T iff A is a palindrome," holds. Note that we have only used part of the loop invariant, that is we used the fact that after the k-th iteration, $i = k + 1$; it still holds that after the k-th iteration, for $1 \leq j \leq k$, $A[j] = A[n - j + 1]$, but we do not need this fact in the above proof.

3. Do problem 2.22 in the notes (which is related the part A, as it requires a proof of correctness of the Gram-Schmidt algorithm).

   Solution: We are going to prove a loop invariant on the outer loop of algorithm 2.6, that is, we are going to prove a loop invariant on the for-loop (indexed on i) that starts on line 2 and ends on line 7. Our invariant consists of two parts: after the k-th iteration of the loop, the following two statements hold true:

   (a) the set $\{v_1^*, \ldots, v_{k+1}^*\}$ is orthogonal, and

   (b) $\text{span}\{v_1, \ldots, v_{k+1}\} = \text{span}\{v_1^*, \ldots, v_{k+1}^*\}$.

   Basis Case: after zero iterations of the for-loop, that is, before the for-loop is ever executed, we have, from line 1 of the algorithm, that $v_1^* \longleftarrow v_1$, and so the first statement is true because $\{v_1^*\}$ is orthogonal (a set consisting of a single non-zero vector is always orthogonal—and $v_1^* = v_1 \neq 0$ because the assumption (i.e., pre-condition) is that $\{v_1, \ldots, v_n\}$ is linearly independent, and so none of these vectors can be zero), and the second statement also holds trivially since if $v_1^* = v_1$ then $\text{span}\{v_1\} = \text{span}\{v_1^*\}$.

   Induction Step: Suppose that the two conditions hold after the first k iterations of the loop; we are going to show that they continue to hold after the $k+1$ iteration. Consider:

$$v_{k+2}^* = v_{k+2} - \sum_{j=1}^{k+1} \mu_{(k+1)j} v_j^*,$$

which we obtain directly from line 6 of the algorithm; note that the outer for-loop is indexed on i which goes from 2 to n, so after the k-th execution of line 2, for $k \geq 1$, the value of the index i is $k+1$. We show the first statement, i.e., that $\{v_1^*, \ldots, v_{k+2}^*\}$ are orthogonal. Since, by IH, we know that $\{v_1^*, \ldots, v_{k+1}^*\}$ are already orthogonal, it is enough to show that for $1 \leq l \leq k+1$, $v_l^* \cdot v_{k+2}^* = 0$, which we do next:

$$\begin{aligned} v_l^* \cdot v_{k+2}^* &= v_l^* \cdot \left(v_{k+2} - \sum_{j=1}^{k+1} \mu_{(k+2)j} v_j^* \right) \\ &= (v_l^* \cdot v_{k+2}) - \sum_{j=1}^{k+1} \mu_{(k+2)j} (v_l^* \cdot v_j^*) \end{aligned}$$

and since $v_l^* \cdot v_j^* = 0$ unless $l = j$, we have:

$$= (v_l^* \cdot v_{k+2}) - \mu_{(k+2)l} (v_l^* \cdot v_l^*)$$

and using line 4 of the algorithm we write:

$$= (v_l^* \cdot v_{k+2}) - \frac{v_{k+2} \cdot v_l^*}{\|v_l^*\|^2} (v_l^* \cdot v_l^*) = 0$$

where we have used the fact that $v_l \cdot v_l = \|v_l\|^2$ and that $v_l^* \cdot v_{k+2} = v_{k+2} \cdot v_l^*$.

For the second statement of the loop invariant we need to show that

$$\text{span}\{v_1, \ldots, v_{k+2}\} = \text{span}\{v_1^*, \ldots, v_{k+2}^*\}, \tag{1}$$

assuming, by the IH, that $\text{span}\{v_1, \ldots, v_{k+1}\} = \text{span}\{v_1^*, \ldots, v_{k+1}^*\}$. The argument will be based on line 6 of the algorithm, which provides us witht he following equality:

$$v_{k+2}^* = v_{k+2} - \sum_{j=1}^{k+1} \mu_{(k+2)j} v_j^*. \tag{2}$$

Given the IH, to show (1) we need only show the following two things:

(a) $v_{k+2} \in \text{span}\{v_1^*, \ldots, v_{k+2}^*\}$, and

(b) $v_{k+2}^* \in \text{span}\{v_1, \ldots, v_{k+2}\}$.

Using (2) we obtain immediatelly that $v_{k+2} = v_{k+2}^* + \sum_{j=1}^{k+1} \mu_{(k+2)j} v_j^*$ and so we have (1). To show (2) we note that

$$\text{span}\{v_1, \ldots, v_{k+2}\} = \text{span}\{v_1^*, \ldots, v_{k+1}^*, v_{k+2}\}$$

by the IH, and so we have what we need directly from (2).