PRINCE OF SONGKLA UNIVERSITY

FACULTY OF ENGINEERING

Final Examination: Semester 2 **Academic Year**: 2003

Date: 21 February 2004 **Time**: 9.00 - 12.00

Subject Number: 240-204 **Room**: R300

Subject Title: Data Structures and Computer Programming Techniques

---

Exam Duration: 3 hours

This paper has 5 pages and 5 questions. (Total marks:120)

Authorized Materials:

- Writing instruments (e.g. pens, pencils).
- Books (e.g. dictionaries) are permitted.
- Calculators are NOT permitted

Instructions to Students:

- *Use only English in programming codes.*
- Attempt all questions.
- Write your answers in an answer book.
- Start your answer to each question on a new page if possible
- Clearly number your answers.
- Any unreadable parts will be considered wrong.

When writing programs, use good layout, and short comments;
marks will not be deducted for minor syntax errors.

ทุจริตในการสอบ

โทษขั้นต่ำ ปรับตกในรายวิชานั้นและพักการเรียน 1 ภาคการศึกษา

โทษสูงสุด ให้ออก

Name_____Code_____

Section_____

Question 1 (35 marks; 40 minutes)

Write a program that creates a linked list of 10 characters, then creates a copy of the list in reverse order.

Use the following data structure:

```
struct listNode {
  char data;
  struct listNode *nextPtr;
};
typedef struct listNode LISTNODE;
typedef LISTNODE *LISTNODEPTR;
```

You may assume that the following functions have *already* been defined:

a) int *isEmpty*(LISTNODEPTR sPtr) that checks if the list is empty

b) void *printList*(LISTNODEPTR sPtr) that print the list.

c) void *insert*(LISTNODEPTR *sPtr, char value) that inserts a new value (a node) into the list in sorted order.

d) void *push*(LISTNODEPTR *sPtr, char value) that inserts a new value (a node) at the stack top.

1.1 Write the following function LISTNODEPTR *reverseList* (LISTNODEPTR a)

that creates a list in the reverse order of the list argument. Hint Use a stack to help in reversing data. Also write a code fragment (ส่วนของโปรแกรม) showing how to call the function. (15 marks)

The example of result of the first function (*reverseList*) is as follows.

List is: A B C D E F G H I J
The list in reverse is: J I H G F E D C B A

Name_____Code_____

Section_____

1.2 Write the following function LISTNODEPTR *merge* (LISTNODEPTR a, LISTNODEPTR b) that merges two sorted lists and returns the merged list. Also write a code fragment (ส่วนของโปรแกรม) showing how to call the above function.

(20 marks)

The example of result of the second function (*merge*) is as follows.

1st list is: A C D G J
2nd list is: B E F H I K
The merged list is: A B C D E F G H I J K

Name_____Code_____

Section_____

Question 2 (5 marks; 10 minutes)

From Question 1, suppose that we don't use a stack for creating a reversed list. Write a recursive function, void *printListBackwards* (LISTNODEPTR a), that recursively prints a list backwards or in reverse order, without creating a new list. Also write a code fragment showing how to call the function.

Hint: Take the approach of the functions preOrder, inOrder or postOrder as an example.

Name_____Code_____

Section_____

Question 3

The function void *outputTree*(TREENODEPTR treePtr, int spaces) displays an input tree on the screen. (40 marks; 40 minutes)

a) Write the output display of this function if the tree input is a binary search tree contains the data (input in the following order): 41 22 100 130 72 145 63 12 145 12.

(10 marks; 20 minutes)

```
void outputTree(TREENODEPTR, int spaces)
{
 int loop;
 while (treePtr != NULL) {
   outputTree(treePtr->rightPtr, spaces + 5);
   for (loop = 1; loop <= spaces; loop++)
    printf(" ");
  printf("%d\n", treePtr->data);
  outputTree(treePtr->leftPtr, spaces + 5);
  treePtr = NULL;
 }
}
```

b) Calculate the size of the tree and how many times the recursive function *size* would be called for getting the answer. (10 marks, 10 minutes)

The function size was called_____times.

The size of the tree is _____.

Name_____Code_____

Section_____

c) Traverse the above tree using *preorder*, *inorder* and *postorder* tree traversal methods. Show the order in which the nodes are visited for **all** 3 methods.

(15 marks, 15 minutes)

Preorder

_____

Inorder

_____

Postorder

_____

c) Show the tree after node *100* has been deleted. (5 marks, 5 minutes)

Question 4

(30 marks; 40 minutes)

a) Write the function int *partition*(int a[], int left, int right) of Quicksort that uses the **middle** element in the array as a pivot. You may assume that function void *swap*(int a[], int x, int y) have *already* been defined (15 marks; 20 minutes)

Hint: The function returns the correct position of the pivot.

Name_____Code_____

Section_____

b) Apply the following sorting algorithms to sort the unordered data:

4 5 1 6 2 3 (When sorted, A is at the lefttmost position.) (15 marks; 20 minutes)

- Selection sort
- Insertion sort
- (Better) Bubble sort

Write out the execution of each algorithm until the algorithm stops. Write in the same pattern as this example:

Unordered data:	3 5 2 7 4 2
End 1st Pass:	3 2 5 4 2 7
End 2nd Pass:	2 3 4 2 5 7
End 3rd Pass:	2 3 2 4 5 7
End 4th Pass:	2 2 3 4 5 7
End 5th Pass:	2 2 3 4 5 7
Stop.	

- Selection sort

Unordered data: 4 5 1 6 2 3

End 1st Pass: _____

_____

_____

_____

_____

_____

_____

_____

_____

Name_____Code_____

Section_____

- Insertion sort

Unordered data: 4 5 1 6 2 3

End 1st Pass: _____

_____

_____

_____

_____

_____

_____

_____

- (Better) Bubble sort

Unordered data: 4 5 1 6 2 3

End 1st Pass: _____

_____

_____

_____

_____

_____

_____

_____

Question 5 (10 marks; 10 minutes)

For the following sorted list:

1 3 6 7 10 12 15 17 19 22 23

Show how to search for 7 and 11 by

a) Sequential Search

b) Binary Search.

Write the considered data in each round.

a)

For 7

_____

For 11

_____

b)

For 7

_____

For 11

_____

--- End of Examination ---

Name_____Code_____

Section_____