Data Structure Midterm Exam

이름 :

학번 :

컴파일 에러 해결을 위한 헤더파일(pch.h) 외의 헤더는 사용이 불가합니다.

블랙보드에서 중간고사 파일을 다운받아 작성 후 학번.zip 파일로 제출해주세요.

.cpp 혹은 .c 파일로 바꾸어 작성해도 무방합니다.

시험지는 이름과 학번을 적어 퇴실시 제출해주세요.

1. fibonacci 수열의 n번째 값을 리턴하는 함수를 작성하시오.

fibonacci(1) = 1
fibonacci(2) = 1
fibonacci(n) = fibonacci(n-1) + fibonacci(n-2)

main.c

```
# include "stdio.h"

long fib_value[90];

long long fibonacci(int n) {

    // Student's Answer

}

int main() {
    for (int i = 0; i < 90; i++) {
        fib_value[i] = -1;
    }
    fib_value[0] = 1;
    fib_value[1] = 1;
    for (int i = 1; i <= 90; i++) {
        printf("%d : %lld\n", i, fibonacci(i));
    }
}
```

출력결과

```
1 : 1
2 : 1
3 : 2
4 : 3
5 : 5
6 : 8
7 : 13
8 : 21

 ... 중략 ...

88 : 1100087778366101931
89 : 1779979416004714189
90 : 2880067194370816120
```

2. String이 올바른 괄호 쌍을 포함하는지 확인하는 함수를 구현하시오.

main.c

```
# include "stdio.h"
# include "stackADT.h"

/*Return 1 if expression has balanced Parenthesis */
bool bracecheck(char exp[]) {

    // Student's answer
}

int main() {
    FILE *pFile = NULL;
    int result;
    pFile = fopen("text.txt", "r");
    if (pFile != NULL) {
        char strTemp[255];
        char *pStr;

        while (!feof(pFile)) {
            pStr = fgets(strTemp, sizeof(strTemp), pFile);
            result = bracecheck(pStr);
            if (result == 1) printf(" %s\n", "pass");
            else printf(" %s\n", "fail");
        }
        fclose(pFile);
    } else {
        printf("%s", "File not found.\n");
    }
    return 0;
}
```

출력결과

```
pass
fail
fail
pass
fail
pass
fail
pass
pass
pass
```

3. listADT에서 insert, exchange, sort 함수를 작성하세요.

listADT.c

```
#include "stdlib.h"
#include "stdio.h"
#include "listADT.h"

typedef struct node {
    char key;
    struct node *prev;
    struct node *next;
} node;

typedef struct list {
    int count;
    node *head;
} list;

list *createList() {
    list *l = (list *) malloc(sizeof(list));
    l->count = 0;
    l->head = NULL;
    return l;
}

node *createNode(char value) {
    node *n = (node *) malloc(sizeof(node));
    n->key = value;
    n->prev = NULL;
    n->next = NULL;
    return n;
}

int isEmptyList(list *l) {
    if (l->head == NULL)
        return 1;
    else
        return 0;
}

node *last(list *l) {
    node *n = l->head;
    while (n->next != NULL)
        n = n->next;
    return n;
}

void pushFirst(list *l, node *n) {
    if (isEmptyList(l)) {
        l->head = n;
    } else {
```

```
        n->next = l->head;
        l->head->prev = n;
        n->prev = NULL;
        l->head = n;
    }
    l->count++;
}

void pushLast(list *l, node *n) {
    if (isEmptyList(l))
        l->head = n;
    else {
        n->prev = last(l);
        last(l)->next = n;
    }
    l->count++;
}

node *nthNode(list *l, int n) {
    node *nd = l->head;
    for (int i = 0; i < n; i++) {
        nd = nd->next;
    }
    return nd;
}

void printList(list *l) {
    if (isEmptyList(l) == 1) {
        printf("%s", "empty\n");
        return;
    }
    printf("length %d : ", l->count);
    for (node *n = l->head; n != NULL; n = n->next) {
        printf("%c ", n->key);
    }
    printf("\n");
}

void insert(list *l, int pos, node *n) {

    // Student's answer

}

void exchange(node *n1, node *n2) {

    // Student's answer

}

void sort(list *l) {

    // Student's answer

}
```

출력결과

```
length 7 : g e d b a c f
insert h 0
length 8 : h g e d b a c f
insert i 4
length 9 : h g e d i b a c f
exchange 3 5
length 9 : h g e b i d a c f
exchange 2 3
length 9 : h g b e i d a c f
sort
length 9 : a b c d e f g h i
```

4. inorder, preorder, postorder 순환 함수를 작성하세요.

binary_tree_ADT.cpp

```
#include<stdio.h>
#include<stdlib.h>
#include "binary_tree_ADT.h"

typedef struct TreeNode {
    char value;
    struct TreeNode *left;
    struct TreeNode *right;
} TreeNode;

TreeNode *createTree(TreeNode *left, char item, TreeNode *right) {
    TreeNode *pNewNode = (TreeNode *) malloc(sizeof(TreeNode));
    if (pNewNode == NULL) return NULL;
    pNewNode->value = item;
    pNewNode->left = left;
    pNewNode->right = right;
    return pNewNode;
}

void traversal_innode(TreeNode *node) {

    // Student's Answer

}

void traversal_prenode(TreeNode *node) {

    // Student's Answer

}

void traversal_postnode(TreeNode *node) {

    // Student's Answer

}
```

출력결과

```
a * b + c + d
+ * a + b c d
a b c + * d +
```