2021 자료구조 수시시험(2021.10.21)

학번: _____ **이름:** _____

1. 다음 주어진 함수 f의 최악의 경우 점근적 시간 복잡도를 O표기법으로 쓰시오.(각 5점)

1)

```
int f(int a[], int n) {
    if (n<=0) return 1;
    else
        return a[0]*f(a+1, n-1)*f(a+1,n-1);
}
```

2)

```
int g(int a[], int n) {
     int s = 1;
     int i;
     for (i=0; i<n; i=i+1)
          s*=a[j];
     return s;
}
int f(int a[], int n) {
    int s = 0;
    int i;
    for (i=0; i<n; i++)
        s += g(a+i*n, n);
}
```

2. 다음 프로그램의 출력을 쓰시오.(컴파일 warning은 무시) (10점)

```
#include <stdio.h>
int foo(int x[3][8]) {
    for (int i=0; i<3; i++) x[i][0] *= 10;
    return **(x+2);
}
int main(){
    int a[4][6] = {{1,2,3,4,5,6},{7,8,9,10,11,12},
            {13,14,15,16,17,18},{19,20,21,22,23,24}};
    int *p = a[1]+2;
    char *q = (char *)(a+1);
    printf("%d\n", (a+1)[1][3]);
    printf("%d\n", *(((int *)q)+sizeof(int)));
    printf("%d\n", *p);
    printf("%d\n", foo(a));
    printf("%d\n", a[1][0]+a[1][1]+a[1][2]);
}
```

3. 다음 각 차원의 크기가 모두 같은 3차원 배열과 한 차원의 원소의 개수 n을 인자로 받아 대각선 위치의 원소(x[i][i][i], i=0,1,..,n-1)의 합을 반환하는 함수 sum3를 완성하시오.(10점)

실행결과
9
45

```
#include <stdio.h>
int sum3(int *a, int n) {
    int i;




    return s;
}
int main() {
    int a[2][2][2] = {{{1,2},{3,4}},{{5,6},{7,8}}};
    int b[3][3][3] = {{{1,2,3},{4,5,6},{7,8,9}},
                      {{11,12,13},{14,15,16},{17,18,19}},
                      {{21,22,23},{24,25,26},{27,28,29}}};
/* a[0,0,0]+a[1,1,1]*/
    printf("%d\n", sum3(&a[0][0][0],2));
/* b[0,0,0]+b[1,1,1]+b[2,2,2]*/
    printf("%d\n", sum3(&b[0][0][0],3));
}
```

4. 다음 문맥 자유 문법으로 정의된 문자열 집합 <myStr>에 대하여 답하시오. (ε는 빈 문자열“”을 나타냄)

<myStr> → ε | ∗ | +***<myStr>***+ | -***<myStr>***-

1) 위 <myStr> 문자열 집합에 속하는 문자열 중 길이가 5인 것을 4가지 제시하시오.(4점)

2) 문자열과 문자열의 길이를 받아서 위 문법에 맞는 문자열이면 1을 아니면 0을 반환하는 재귀 함수 isMyStr을 작성하시오. (6점)

```
int isMyStr(char *a, int n) {




}
main() {
    printf("%d %d", isMyStr("+-",2), isMyStr("*",1));
}
```

(실행결과)
0 1

5. 다음 코드를 아래와 같이 완성하시오.

실행결과
1 2 3 4 5 6 7 8 9 10 11
2 4 6 8 10

```
#include <stdio.h>
#include <stdlib.h>
typedef struct _node *nodeptr;
typedef struct _node {
     int key;
     nodeptr next;
} node;
node* init_list(void){
     node *head = (node*)malloc(sizeof(node));
     node *tail = (node*)malloc(sizeof(node));
     head->next = tail;
     tail->next = tail;
     return head;
}
void printList(nodeptr h) {
     /* 리스트의 원소드를 순서대로 출력한다. (코드 생략) */
     /*  ...     */
}
void deleteOddNums(nodeptr h) {









}
node *makelist(int n) {
     node *h;









     return h;
}
int main() {
    nodeptr p;
    p = makelist(11);
    printList(p);
    deleteOddNums(p);
    printList(p);
}
```

1) 정수n을 받아, 1부터 n까지의 숫자로 이루어진 단순 연결리스트를 만들어 그 헤드노드를 반환하는 함수 makelist를 작성하시오.(반드시 반복문을 사용하시오. 위 프로그램에서는 1부터 11까지로 이루어진 단순 연결리스트를 만든다)(8점)

2) 리스트 내에서 홀수값을 가진 노드를 모두 없애는 함수 deleteOddNodes를 작성하시오.(7점)

6. 다음 postfix 함수는 중위 표기법 식을 후위 표기법으로 변환하는 함수이다. 이에 대하여 물음에 답하시오.

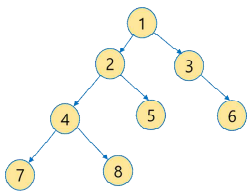
```
void postfix(char *dst, char *src){
    char c;
    init_stack(); // 스택 초기화
    while (*src) {
      if (*src== '(') {  push(*src); src++; }
      else if (*src== ')') {
          while (get_stack_top() != '(') {
             *dst++ = pop(); *dst++ = ' ';
          } pop(); src++;
      }
      else if (is_operator(*src)) {
          while (!is_stack_empty() &&
           precedence(get_stack_top())>= precedence(*src)){
            printf("%c ",get_stack_top());
            *dst++ = pop(); *dst++ = ' ';
          }
          printf("\n");
          push(*src); src++;
      }
      else if (*src>= '0' && *src<= '9') {
          do {
              *dst++ = *src++;
          } while (*src>= '0' && *src<= '9');
          *dst++ = ' ';
      }
      else src++; }
    while (!is_stack_empty()) { *dst++ = pop(); *dst++ = ' '; }
    dst--; *dst= 0;
}
```

1) "2+3*(4/2-1+3)/2+3"을 인자로 전달할 때 위 밑줄 그은 두 개의 printf문으로 인한 출력을 모두 정확히 쓰시오.(5점)

2) 위 함수의 최악의 경우 메모리 복잡도를 제시하고 그 근거를 설명하시오.(5점)

7. 다음 함수를 사용하여 주어진 트리를 순회하려 한다.

```
void traverse(node *t) {
  init_stack(); push(t);
  while ((t=pop()) != NULL) {
    visit(t);
    if (t->right != tail) push(t->right);
    if (t->left != tail) push(t->left);
  }
}
```



1) 위 함수를 사용하여 다음의 트리를 순회한다고 할 때 7 노드 방문 시점의 스택에 저장된 노드의 값을 스택 탑부터 순서대로 쓰시오.(5점)

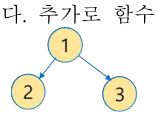
2) 위 프로그램과 같은 동작을 하는 재귀 함수를 작성하시오.(5점)

8. 다음 트리를 만드는 프로그램에 대하여 물음에 답하시오.

```
typedef struct_node {
    int key;
    struct _node *left;
    struct _node *right;
} node;
node *head, *tail;
void init_tree(void) {
    head = (node*)malloc(sizeof(node));
    tail = (node*)malloc(sizeof(node));
    head->left = tail;
    head->right = tail;
    tail->left = tail;
    tail->right = tail;
}
```

```
int sumTree() {




}
main() {
   init_tree();









   printf("%d\n" sumTree());
}
```

1) 다음 모양의 트리를 만드는 코드를 main 함수에 작성하시오.(루트 노드는 head노드의 오른쪽에 연결되고, 리프 노드는 자식으로 tail노드를 가리킨다. 추가로 함수를 정의하여 사용해도 됨)(8점)



2) 트리 내 노드들의 값을 합산하는 함수 sumTree를 작성하시오. (1),2)를 완료한 프로그램은 6을 출력함) (7점)

9. 다음 함수에 대하여 물음에 답하시오.

```
#define  MaxCut 5
int price[] = {0,1,5,8,9,10};
int maxprices[101] = {0};    /* 2)번을 위해 사용 */
int *cut;
int maxp(int m) {
    int i,p;
    int max;
    int maxCut = MaxCut;
    int t = 0;
    if (m<=0) return 0;
    if (m<MaxCut) maxCut = m;
    max = 0;
    for (i=1; i<=maxCut; i++) {
        p = price[i]+maxp(m-i);
        if (p>max) {
            max = p;
        }
    }
    return max;
}
int main() {
    int i, len;
    printf("Length of stick? ");
    scanf("%d", &len);
    int p = maxp(len);
    printf("%d\n",p);
}
```

1) maxp 함수의 의미와 입력 6에 대한 출력을 쓰시오. (5점)

2) 위 maxp 함수를 dynamic programming 기법을 사용하여 O(n) 시간 복잡도를 가지는 함수로 고치시오.(위 코드에 적절히 수정한다. 최대 입력값은 100으로 가정, 5점)

2020 자료구조 수시시험(2020.10.21)

학번: **이름:**

1. 다음 프로그램의 최악의 경우 시간 복잡도를 O표기법으로 쓰시오.(15점)

1)

```
int foo1(int a[], int n) {
    int i;
    int s = 0;
    for (i=1; i<=n; i=i*2) s+=a[j];
    return s;
}
```

2)

```
int foo2(int a[], int n) {
   if (n<=0) return 0;
   else
      return a[0] + foo(a+2, n-2);
}
```

3)

```
int foo3(int a[], int n) {
    int i, j;
    int s = 1;
    for (i=1; i<=n; i=i+1)
       for (j=0; j<i; j++)
          s*=a[j];
    return s;
}
```

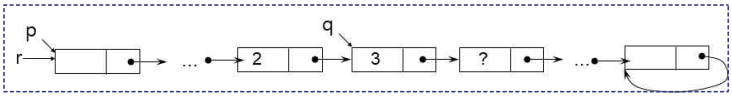
2. 다음 프로그램의 출력을 쓰시오.(10점)

```
#include <stdio.h>
void foo(int *p) {
   int i,j;
   for (i=0;i<=1;i++)
      for (j=2; j<=4;j++)
         printf("%d ", p[i*6+j]);
}
int main() {
    int i, j, k;
    int m[5][3]={{0,1,2},{3,4,5},{6,7,8},{9,10,11},{12,13,14}};
    int *p = (int *)(m+2);
    printf("%d\n", *p);
    printf("%d\n", (m+2)[1][2]);
    printf("%d\n", *(m[1]+4));
    foo(&(m[0][0]));
    return 0;
}
```

3. 다음 프로그램의 출력을 제시하시오.(10점)

```
void foo3(int a[][3], int b[][3]) {
  int n=a[0][1]; int terms = a[0][2];
  int *s = (int*)calloc(n,sizeof(int));
  int *t = (int*)calloc(n,sizeof(int));
  int i,j;
  b[0][0]=n; b[0][1]=a[0][0]; b[0][2]=terms;
  for (i=1; i<=terms; i++) s[a[i][1]]++;
  t[0] = 1;
  for (i=1; i<n; i++) t[i]=t[i-1]+s[i-1];
  for (i=0; i<n; i++) printf("%d ", t[i]);
  printf("\n");
  for (i=1; i<=terms; i++) {
     j = t[a[i][1]];
     b[j][0] = a[i][1];
     b[j][1] = a[i][0];
     b[j][2] = a[i][2];
     t[a[i][1]] = j+1;
  }
}
int main() {
   int a[5][3] = {{4,5,4},{1,2,3},{2,0,4},{3,1,5},{3,0,6}};
   int b[5][3];
   int i,j;
   foo3(a,b);
   for (i=0; i<3; i++)
      for (j=0; j<4; j++)
         printf("%d ", b[i][j]);
}
```

4. 다음 코드에서 main 함수 내에서 q=randomNodeInList(p)까지 수행을 완료한 시점의 단순 연결리스트의 상태가 다음 그림과 같다고 하자.이에 대하여 다음과 같이 추가적인 코드를 작성하시오.(15점)




```
typedef struct _node *nodeptr;
typedef struct _node {
     int key;
     nodeptr next;
} node;
int sum_list(nodeptr p) {   // 2)번 작성









}
main() {
    nodeptr p, q, r;
  //    r = p = makeSomeList();  // 그림의 연결 리스트 만듬
 //   q = randomNodeInList(p);  // 3 노드를 반환
    // 이 시점의 리스트 상태가 위의 그림과 같음.
    printf("sum of keys", sum_list(p));
  // 1)번 코드가 들어갈 곳
  // 3)번 코드가 들아갈 곳
}
```

1) q 다음 노드의 key값(?)을 출력하는 printf문을 작성하시오.(q 변수 사용)

2) 단순 연결 리스트의 원소들의 합을 계산하는 함수인 sum_list(p)를 작성하시오.(헤드 노드의 값은 더하지 않음. 위 코드에 추가하시오)

3) q가 가리키고 있는 노드를 연결리스트에서 삭제하는 코드를 작성하시오. (결과적으로 2 노드 다음 노드가 ?노드가 됨)

5. 중위식 “3 / (4 + 5 * 6) + 7”에 대하여 물음에 답하시오. (20점)

1) 위 식을 전위식으로 바꾸시오.

2) 위 식에 대한 수식 나무(parse tree)를 그리시오.(수식 나무에 괄호를 표기할 필요는 없음)

3) 위 수식 나무를 배열을 사용한 트리 저장(표현)법을 사용하여 배열에 저장한 결과를 그리시오.

4) 위 중위식을 다음 함수를 사용하여 후위식으로 변환하려 한다. 스택의 변화 단계를 모두 제시하시오.(예: 1 -> + 1 -> 2 + 1 -> + 1...)

```
void postfix(char *dst, char *src){
    char c;
    init_stack(); // 스택 초기화
    while (*src) {
      if (*src== '(') {  push(*src); src++; }
      else if (*src== ')') {
          while (get_stack_top() != '(') {
             *dst++ = pop(); *dst++ = ' ';
          } pop(); src++;
      }
      else if (is_operator(*src)) {
          while (!is_stack_empty() &&
              precedence(get_stack_top())>= precedence(*src)){
            *dst++ = pop(); *dst++ = ' ';
          }
          push(*src); src++;
      }
      else if (*src>= '0' && *src<= '9') {
          do {
              *dst++ = *src++;
          } while (*src>= '0' && *src<= '9');
          *dst++ = ' ';
      }
      else src++; }
    while (!is_stack_empty()) { *dst++ = pop(); *dst++ = ' '; }
    dst--; *dst= 0;
}
```

6. 다음은 큐를 구현하여 사용하는 프로그램이다.(10점)

```
#define MAX 10
typedef struct _node {
    int front;
    int rear;
    int data[MAX];
} queue;
queue *create_queue() {
   queue *q = (queue *)malloc(sizeof(queue));
   q->front = q->rear = 0;
   return q;
}
int put(queue *q, int k) {
    if ((q->rear + 1) % MAX == q->front) {
        printf("\n      Queue overflow."); return -1;
    }
    q->data[q->rear] = k;
    q->rear = ++q->rear % MAX;
    return k;
}
```

```
int get(queue *q){




}
void print_queue(queue *q)  {
    int i;
    for (i= q->front; i!=q->rear; i= ++i% MAX)
        printf("%-6d", q->data[i]);
}
void main(void){
    int i;
    queue *q = create_queue();
    put(q, 1); put(q, 2); put(q, 3);
    get(q); get(q);
    put(q,4); put(q, 5); put(q, 6); put(q, 7);
    get(q); get(q); get(q);
    put(q,8); put(q, 9); put(q, 10); put(q, 11); put(q, 12);
    print_queue(q);
}
```

1) 위 프로그램의 get 함수를 완성하시오.

2) main 함수의 print_queue 호출 직전의 힙 메모리 상태를 큐 자료구조를 중심으로 자세히 그리시오. (q변수가 가리키는 큐 구조체 내의 모든 필드값들의 상태(배열 내 값 포함)를 정확히 명시하시오.)

7. 다음은 피보나치 수열 계산 프로그램이다. 동적 프로그래밍 기법을 사용하여 fibo함수에 문장을 추가하여, 함수의 점근적 시간복잡도를 O(n)으로 향상시키시오. (입력은 1이상 45이하로 가정함. 수정 후에도 fibo는 계속 재귀함수여야 함. main 함수는 수정하지 않음)(10점)

```
#define MAX 46
int fibos[MAX] = {0};  // 모두 0으로 초기화
int fibo(int n) {
        int fib;



        if (n==1 || n==2)
                fib = 1;
        else
                fib = fibo(n-1)+fibo(n-2);



    return fib;
}
main() {
   int n;
   scanf("%d", &n);
   if (n<1) {
      printf("Number must be positive!");  return 0;
   }
   printf("%d th fibonacci number is %u\n",n , fibo(n));
}
```

8. 다음은 길이 n의 배열 세 개를 받아 두 배열의 대응하는 원소의 합을 세 번째 배열에 저장하는 함수이다. 이 프로그램을 재귀 함수로 작성하시오. (10점)

```
void pair_sum(int a[], int b[], int s[], int n) {
    int j = 0;
    for (j=0; j<n; j++) s[j] = a[j] + b[j];
    return;
}
```

2019 자료구조 수시시험('08.4.30)
(Closed Book)

1. 다음 함수들의 점근적 시간 복잡도를 계산하시오.(O표기법 사용) (각 5점)

1)
```
int foo1(int a[], int n) {
      int i, j;
      int s = 0;
      for (i=1; i<=n; i=i+1) {
         for (j=0; j<i; j++)
            s+=a[j];
      }
}
```

2)
```
int foo1(int a[], int n) {
      int i, j;
      int s = 0;
      for (i=1; i<=n; i=i*2) {
         for (j=0; j<i; j++)
            s+=a[j];
      }
}
```

3)
```
int foo2(int a[], int n) {
   if (n == 0 || a[0] == 0)
      return 1;
   return (a[0] * foo2(a+1, n-1));
}
```

2. 다음 프로그램의 실행 결과를 쓰시오.(10점)
```
#include <stdio.h>
int main() {
    int a[3][4] = {{1,2,3,4},{5,6,7,8},{9,10,11,12},{13,14,15,16}};
    int *b;
    int *p;

    b = a[1];
    p = (int *)a;
    printf("%d\n", b[2]);
    printf("%d\n", **(a+2));
    printf("%d\n", (a+1)[1][2]);
    printf("%d %d\n", sizeof(a[1]), sizeof(p[1]));
    printf("%d %d\n", *(int*)(a+2), *(p+2));
}
```

3. 다음은 sparse matrix를 보기 좋게 출력하고, (1,2), (2,1) 위치의 값을 출력하는 프로그램이다(value 함수는 미완성임). sparse matrix의 0행에는 sparse matrix의 각 차원의 크기와 0이 아닌 원소의 개수가 저장되고, 다음의 각각의 행에는 행 번호, 열 번호, 정수 데이터가 저장된다.
```
#include <stdio.h>

int value(int mat[][3], int i, int j) {
    int v = 0;

    return v;
}

int main() {
      int i, j, k;
      int mat[][3] = {{3,4,4},{0,0,3},{1,3,5},{2,1,8},{2,3,7}};
      k = 1;
      for (i=0; i<mat[0][0]; i++) {
         for (j=0; j<mat[0][1]; j++)
            if ((k<=mat[0][2]) && (mat[k][0]==i)
               && (mat[k][1]==j))
                  printf("%d ", mat[k++][2]);
            else
                  printf("0 ");
            printf("\n");
         }

         printf("mat[%d][%d]: %d\n", 1, 2, value(mat, 1, 2));
         printf("mat[%d][%d]: %d\n", 2, 1, value(mat, 2, 1));
}
```

1) 위 프로그램의 출력을 쓰시오. (5점)
2) value 함수가 주어진 sparse matrix의 i, j 위치에 저장된 값을 반환하도록 수정하여, 밑줄 그은 두 printf 문의 출력 결과가 다음과 같이 되도록 하시오.
```
mat[1][2]: 0
mat[2][1]: 8
```

4. 다음 프로그램에 대하여 물음에 답하시오.
```
sum3(int a[][3]) {
     int i;
     int s = 0;
     for (i=0; i<3; i++) s += a[i][i];
     return s;
}

sum4(int a[][4]) {
     int i;
     for (i=0; i<4; i++) s += a[i][i];
     return s;
}

main() {
     int i;
     int a[3][3] = {{0,1,2},{3,4,5},{6,7,8}};
     int b[4][4] = {{0,1,2,3},{4,5,6,7},{8,9,10,11},{12,13,14,15}};
     printf("%d\n", sum3(a));
     printf("%d\n", sum4(b));
}
```

1) 위 프로그램의 출력을 쓰시오. (5점)
2) 임의의 n*n 배열에 대하여 sum3.함수 및 sum4 함수의 동작을 하도록 sum 함수를 작성하여 두 번의 호출이 모두 sum 함수를 호출하도록 프로그램을 수정하시오. (프로그램의 출력은 같아야 하며, 함수 인자의 타입 및 개수는 수정 가능하다)(10점)

5. 다음 프로그램에 관하여 물음에 답하시오.
```
#include <stdio.h>
#include <stdlib.h>
typedef struct _node {
          int key;
          struct _node *next;
} node;
node *head, *tail;

void init_stack(void) {
          head = (node*)malloc(sizeof(node));
          tail = (node*)malloc(sizeof(node));
          head->next = tail;
          tail->next = tail;
}
int is_stack_empty(void) { return (head->next == tail); }

int get_stack_top() {
          return (is_stack_empty() ? -1 : head->next->key);
}

int push(int k) {
          node *t;
          t = (node*)malloc(sizeof(node));
          if (t == NULL) return -1;
          t->key = k; t->next = head->next; head->next = t;
          return k;
}

int pop(void) {
          node *t;
          int i;
          if (head->next == tail) return -1;
          t = head->next; i = t->key;
          head->next = t->next;
          return i;
}

void clear_stack() {
       head->next = tail;
}
```

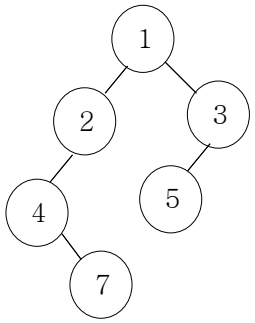
```
main() {
        init_stack();
        push(1);
        push(2);
        push(3);
        printf("%d\n", pop());
        printf("%d\n", pop());
        clear_stack();
        push(1);
        push(2);
        push(3);
        pop();
        push(4);
        printf("%d\n", pop());
        printf("%d\n", pop());
        printf("%d\n", pop());
}
```

1) 위 프로그램의 수행 시 출력을 쓰시오. (5점)
2) 위에서 제시한 스택 함수를 계속해서 동작하는 프로그램이 계속 사용할 경우 메모리 문제가 발생할 수 있다. 어떤 문제가 발생하는지 설명하고, 이러한 문제가 발생하지 않도록 문제가 되는 함수를 모두 수정하시오. (10점)

6. 다음은 중위 표기법을 후위 표기법으로 변환하는 프로그램이다. 위 프로그램의 실행할 경우 출력을 정확히 쓰시오. (생략된 스택 관련 함수는 문제 5번을 참고하시오.)(10점)

```
int is_operator(int k) {
    return (k=='+' || k=='-' || k=='*' || k=='/');
}

int precedence(int op) {
    if (op == '(') return 0;
    if (op == '+' || op == '-') return 1;
    if (op == '*' || op == '/') return 2;
    else return 3;
}
void postfix(char *dst, char *src){
    char c;
    init_stack();
    while (*src) {
        if (*src == '(') {
            push(*src); src++;
        }
        else if (*src == ')') {
            while (get_stack_top() != '(') {
                *dst++ = pop(); *dst++ = ' ';
            }
            pop(); src++;
        }
        else if (is_operator(*src)) {
            while (!is_stack_empty() &&
              precedence(get_stack_top())>=precedence(*src)) {
           c = pop(); *dst++ = c; *dst++ = ' ';
           printf("%c ", c);
            }
            printf("\n");
            push(*src); src++;
        }

        else if (*src >= '0' && *src <= '9') {
            do {
        *dst++ = *src++;
            } while (*src >= '0' && *src <= '9');
            *dst++ = ' ';
        }
        else src++;
    }
    while (!is_stack_empty()) {
            *dst++ = pop(); *dst++ = ' ';
    }
    dst--; *dst = 0;
}

main() {
   char exp[30] = "1 - (2 + 3 * 2 / 2 + 7)";
   char result[30];
   postfix(result, exp);
   printf("%s\n", result);
}
```

7. 다음 트리를 배열을 사용해서 저장하고, 저장한 방법의 원리와 한 노드의 위치에서 부모 및 자식 노드에 접근하는 방법을 설명하시오.(10점)



8. 연결리스트에 x에 대한 다항식을 입력 받아 저장하고, 이에 대한 다음 연산을 수행하는 프로그램을 작성하시오. 각 항은 반드시 mx^n의 형태라고 가정하며, 입력 시에는 항의 개수와 각 항의 차수 및 지수를 입력받으며, 항상 차수가 높은 항부터 낮은 항으로 정렬된 형태로 입력된다고 가정한다. (항상 차수는 0보다 크거나 같음.) (10점+숙제 copy 확인)

(실행화면)

```
> 다항식을 입력하세요: 4 2 5 1 3 -3 1 2 0
-> f(x) = 2x^5 + x^3 - 3x + 2
-----------------------------------------
> 작업을 선택하세요 (더하기:1, 대입 2, 종료 3): 2
> x의 값을 입력하세요: -2
-> f(-2) = -64
-----------------------------------------
> 작업을 선택하세요 (더하기:1, 대입 2, 종료 3): 1
> f(x)에 더할 다항식을 입력하세요: 2 2 2 4 1
-> f(x) = : 2x^5 + x^3 + 2x^2 + x + 2
-----------------------------------------
> 작업을 선택하세요 (더하기:1, 대입 2, 종료 3): 1
> f(x)에 더할 다항식을 입력하세요: 2 -1 1 3 0
-> f(x) = : 2x^5 + x^3 + 2x^2 + 5
-----------------------------------------
> 작업을 선택하세요 (더하기:1, 대입 2, 종료 3): 2
> x의 값을 입력하세요: 2
-> f(2) = 85
-----------------------------------------
> 작업을 선택하세요 (더하기:1, 대입 2, 종료 3): 3
-> Bye
```