11 주차 실습

실습

구현 방법 (인접 행렬)

```c
#include <stdio.h>
#include <stdlib.h>

#define MAX_VERTICES 50
typedef struct GraphType {
  int n;
  int adj_mat[MAX_VERTICES][MAX_VERTICES
} GraphType;

void init(GraphType* g){
  int r, c;
  g->n = 0;
  for (r = 0; r < MAX_VERTICES; r++)
    for (c = 0; c < MAX_VERTICES; c++)
      g->adj_mat[r][c] = 0;
}
void insert_vertex(GraphType* g, int v){
  if (((g->n) + 1) > MAX_VERTICES) {
    fprintf(stderr, "그래프: 정점의
          개수 초과");
    return;
  }
  g->n++;
}
```

```c
void insert_edge(GraphType* g, int start, int end){
  if (start >= g->n || end >= g->n) {
    fprintf(stderr, "그래프: 정점 번호 오류");
  return;
  }
  g->adj_mat[start][end] = 1;
  g->adj_mat[end][start] = 1;
}
void print_adj_mat(GraphType* g){
  for (int i = 0; i < g->n; i++) {
    for (int j = 0; j < g->n; j++) {
      printf("%2d ", g->adj_mat[i][j]);
    }
    printf("₩n");
  }
}
void main(){
  GraphType* g = (GraphType*)malloc(sizeof(GraphType));
  init(g);
  for (int i = 0; i < 4; i++)
    insert_vertex(g, i);
  insert_edge(g, 0, 1);
  insert_edge(g, 0, 2);
  insert_edge(g, 0, 3);
  insert_edge(g, 1, 2);
  insert_edge(g, 2, 3);
  print_adj_mat(g);
  free(g);
}
```

실습

구현 방법 (인접 리스트)

```c
#include <stdio.h>
#include <stdlib.h>
#define MAX_VERTICES 50
typedef struct GraphNode{
  int vertex;
  struct GraphNode* link;
} GraphNode;
typedef struct GraphType {
  int n;
  GraphNode* adj_list[MAX_VERTICES];
} GraphType;
void init(GraphType* g){
  int v;
  g->n = 0;
  for (v = 0; v < MAX_VERTICES; v++)
    g->adj_list[v] = NULL;
}
void insert_vertex(GraphType* g, int v){
  if (((g->n) + 1) > MAX_VERTICES) {
    fprintf(stderr, "그래프: 정점의 개수 초과");
    return;
  }
  g->n++;
}
```

```c
void insert_edge(GraphType* g, int u, int v){
  GraphNode* node;
  if (u >= g->n || v >= g->n) {
    fprintf(stderr, "그래프: 정점 번호 오류");
    return;
  }
  node = (GraphNode*)malloc(sizeof(GraphNode));
  node->vertex = v;
  node->link = g->adj_list[u];
  g->adj_list[u] = node;
}
void print_adj_list(GraphType* g){
  for (int i = 0; i < g->n; i++) {
    GraphNode* p = g->adj_list[i];
    printf("정점 %d의 인접 리스트 ", i);
    while (p != NULL) {
      printf("-> %d ", p->vertex);
      p = p->link;
    }
  printf("₩n");
  }
}
```

실습

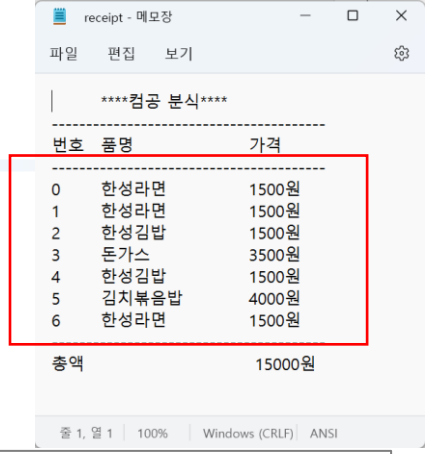
구현 방법 (인접 리스트)

```
int main(){
  GraphType* g =
(GraphType*)malloc(sizeof(GraphType));
  init(g);
  for (int i = 0; i < 4; i++)
    insert_vertex(g, i);
  insert_edge(g, 0, 1);
  insert_edge(g, 1, 0);
  insert_edge(g, 0, 2);
  insert_edge(g, 2, 0);
  insert_edge(g, 0, 3);
  insert_edge(g, 3, 0);
  insert_edge(g, 1, 2);
  insert_edge(g, 2, 1);
  insert_edge(g, 2, 3);
  insert_edge(g, 3, 2);
  print_adj_list(g);
  free(g);
  return 0;
}
```

실습

C 언어 파일 입출력

- 다음 소스는 분식집에서 음식을 주문하고 영수증을 출력하는 프로그램 일부이다. 해당 소스를 설명을 듣고 빈 곳을 개발해보세요




```
#include <stdio.h>
#include <stdlib.h>
void main() {
   char* menu[4][2] = {
      {"한성라면","1500원"},
      {"한성김밥","1500원"},
      {"돈가스","3500원"},
      {"김치볶음밥","4000원"}
   };
   int input, order_num = 0;
   int orders[100] = { 0 }, totalPrice = 0;
   printf("----------------메뉴판----------------Wn");
   printf("0. 주문 종료Wn");
   printf("1. %s   / %s Wn", menu[0][0], menu[0][1]);
   printf("2. %s   / %s Wn", menu[1][0], menu[1][1]);
   printf("3. %s     / %s Wn", menu[2][0], menu[2][1]);
   printf("4. %s / %s Wn", menu[3][0], menu[3][1]);
   printf("--------------------------------------Wn");
   while (1) {
      input 값을 입력 받는다.
      1~4 사이 값인 경우 orders 배열에 -1을 하여 삽입(주문)
      order_num 을 하나 증가
      0일 경우 while 문 종료
      그 외 값일 경우 "없는 메뉴라고 출력"
   }
```

```
   FILE* fp = NULL;
   fopen_s(&fp, "receipt.txt", "wt");
   if (fp != NULL) {
      fprintf(fp, "Wt****컴공 분식****Wn");
      fprintf(fp, "--------------------------------------Wn");
      fprintf(fp, "번호Wt품명WtWtWt가격Wn");
      fprintf(fp, "--------------------------------------Wn");

fprintf를 활용하여 위 그림과 같이 입력한 내용을 파일로 출력

      fprintf(fp, "--------------------------------------Wn");
      fprintf(fp, "총액WtWtWtWt %d원 Wn", totalPrice);
      fclose(fp);
      printf("WnWn주문 내역을 receipt.txt로 출력 하였습니다Wn");
      printf("Wn주문 내역Wn");
   }
```

```
   fopen_s(&fp, "receipt.txt", "r");
   char readStr1[80], readStr2[80];
   if (fp != NULL) {
      for (int i = 0; i < 4; i++) {
         fgets(readStr1, 80, fp);
         printf("%s", readStr1);
      }

   fscanf_s를 활용하여 파일 값 입력 받아 printf 로 출력

      for (int i = 0; i < 3 ; i++) {
         fgets(readStr1, 80, fp);
         printf("%s", readStr1);
      }
   fclose(fp);
}}
```

실습

인접 행렬 파일을 인접 리스트로 변환하기

- 아래와 같은 인접 행렬 형태의 텍스트가 존재할 때, 이를 인접 리스트 그래프로 변환하시오
- 기존 인접 리스트 소스코드 활용
- 해당 텍스트는 파일로 저장 후 불러와 활용
  (이 때, fopen_s, fscanf_s 사용)

```
typedef struct GraphNode {
  int vertex;
  struct GraphNode* link;
} GraphNode;
typedef struct GraphType {
  int n;
  char value[MAX_VERTICES];
  GraphNode* adj_list[MAX_VERTICES];
} GraphType;
```

```
4
A 0 1 0 1
B 1 0 1 1
C 0 1 0 1
D 1 1 1 0
```