

여러분은 디저트와 커피를 판매하는 무인 카페를 운영 중입니다. 무인 카페는 외부업체에 디저트와 커피를 주문해서 재고를 확보하고, 손님에게 판매를 합니다. 여러분은 **“무인 카페 프로그램”**을 이용해서 원격에서 무인 카페를 운영합니다. 이 프로그램은 관리자 모드와 서비스 모드 두 가지 모드로 동작합니다.

- 관리자 모드: 외부업체에 디저트와 커피를 주문해서 재고를 확보
- 서비스 모드: 손님에게 주문을 받고, 영수증을 발급

이제부터 여러분은 4단계에 걸쳐서 무인 카페를 운영하기 위한 **“무인 카페 프로그램”**을 완성하세요. 다음의 사항을 주의하시기 바랍니다.

1. **각 단계의 개발 결과물은 각각 제출되어야 하며, 각 단계의 프로그램은 이전 단계의 결과물을 포함해야 합니다.**
2. **주어진 함수의 반환값과 매개변수는 여러분이 정의하시기 바랍니다.**

[1] 무인 카페 프로그램에서 관리자 모드와 서비스 모드로 진입하는 기능을 개발

무인 카페 프로그램은 로그인 기능을 제공하며, 프로그램 사용자가 입력한 로그인 아이디를 통해 관리자 모드와 서비스 모드를 구분합니다. 로그인 아이디에 따른 각 모드는 다음과 같습니다. (별도의 패스워드는 없습니다.)

- 관리자 모드 아이디: admin
- 서비스 모드 아이디: user

로그인 기능을 구현하기 위해 아래 2개의 함수를 작성하세요.

- main() 함수는 로그인 처리를 위해 check_ID()를 호출하고, 로그인 결과를 바탕으로 현재 프로그램이 어떤 모드로 동작하는지를 사용자에게 알려줍니다.

<로그인 처리 예시>

```
Mode Select (admin or user) : admin  Mode Select (admin or user) : user
Admin mode                          User mode
```

```
Mode Select (admin or user) : 1234
Error : Wrong Input!
```

- check_ID() 함수는 다음과 같은 기능을 처리합니다.
 - 사용자에게 아이디를 입력 받습니다.
 - 등록된 아이디가 아닐 경우 다시 입력 받습니다. 이때, 안내 메시지를 출력합니다.
 - 등록된 아이디일 경우 어떤 모드인지 결과를 main()에게 반환합니다.

[2] 서비스 모드 #1: 주문 메뉴 출력 기능 개발

서비스 모드로 로그인하면 무인 카페 프로그램은 사용자에게 주문 메뉴를 출력하고 판매 서비스를 시작합니다. 본 단계에서는 파일에서 메뉴 정보를 읽어 저장하고, 사용자에게 주문 메뉴를 출력하며, 주문 결과를 알려주는 기능을 구현합니다.

음료수와 스낵은 다음의 파일에 메뉴 정보가 있습니다. 각 파일에는 메뉴명만 존재합니다.

- 음료수: drink1.txt
- 스낵: snack1.txt

주문 메뉴 출력은 단계적으로 이루어져야 합니다. 먼저 음료수와 스낵 중에서 선택을 하고, 다음으로 메뉴를 선택합니다. 또한, 복수 메뉴 주문이 가능하기 때문에 주문 메뉴는 이러한 점을 고려해서 구성되어야 합니다. 다음의 결과 예시를 참조하세요.

<주문 메뉴 예시>

<pre>Mode Select (admin or user) : user ===== drink Menu ===== 1 : americano 2 : caffe latte 3 : ice tea 4 : orange juice 5 : mango juice 6 : cold brew 7 : Peppermint 8 : Earl Gray 9 : Chamomile 10 : green tea 11 : caffe moca 12 : Caramel Macchiato 13 : milk shake 14 : lemonade ===== snack Menu ===== 1 : cream cake 2 : cream bagel 3 : scone 4 : apple pie 5 : cheese cake 6 : tiramisu 7 : egg tart</pre>	<pre>===== snack Menu ===== 1 : cream cake 2 : cream bagel 3 : scone 4 : apple pie 5 : cheese cake 6 : tiramisu 7 : egg tart Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 14 Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 14 Type 1 for drink, 2 for snack, 3 for buy : 2 Type snack id to buy : 7 Type 1 for drink, 2 for snack, 3 for buy : 2 Type snack id to buy : 7 Type 1 for drink, 2 for snack, 3 for buy : 3 ===== Receipt ===== lemonade lemonade egg tart egg tart</pre>
user 모드 접근 시 메뉴 출력	메뉴 출력에 이은 메뉴 주문

Type 1 for drink, 2 for snack, 3 for buy : 4 Error : Wrong input for type select!	Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 15 Error : Not enough ID for drink
drink, snack, buy 중에 선택되지 않을 경우	메뉴 선택 시 id가 없을 경우

주문 메뉴 출력 기능을 위해 다음의 함수들을 구현하세요. 음료 메뉴 정보와 스낵 메뉴 정보 저장을 위해 포인터 배열과 동적 메모리 할당을 사용하세요 (일반 배열 사용 시 50% 감점). 사용자의 주문 메뉴 리스트 저장을 위해서도 포인터 배열과 동적 메모리 할당을 사용해야 합니다 (일반 배열 사용 시 50% 감점, reallocation 사용 시 100% 추가 점수 부여)

- main() 함수는 1단계 기능을 포함하며, 서비스 모드로 로그인 하였을 경우 아래 함수들을 호출하여 주문 메뉴를 출력하고, 주문 결과를 알려줍니다.
- reset_drink_menu() 함수는 drink1.txt 파일에서 음료 메뉴 정보를 읽어서 저장합니다.
- reset_snack_menu() 함수는 snack1.txt 파일에서 스낵 메뉴 정보를 읽어서 저장합니다.
- service_mode() 함수는 화면에 주문 메뉴를 출력하고, 주문을 받으며, 사용자가 주문을 종료하면 주문 메뉴 리스트를 화면에 출력합니다. 주문을 받기 위해서는 각 메뉴의 메뉴번호가 필요합니다. 메뉴가 저장된 배열의 인덱스 번호를 메뉴번호로 활용하세요.

★ 배열의 할당된 크기를 확인하고 싶은 경우 다음의 예시코드를 참고하세요.

```
int main() {
    int* arr = malloc(10 * sizeof(int));

    int allocated_size = _msize(arr);
    printf("Allocated size : %d\n", allocated_size);
    printf("How many indexes? %d\n", allocated_size / sizeof(int));
}
```

[3] 서비스 모드 #2: 재고 관리 및 영수증 발급 기능 개발

2단계에서 우리가 개발한 프로그램은 각 메뉴의 재고를 고려하지 않았습니다. 3단계에서는 각 메뉴의 재고를 고려하여 사용자로부터 주문을 받도록 서비스 모드를 확장합니다. 그리고, 각 메뉴의 가격 정보를 이용하여 사용자에게 영수증을 발급합니다.

각 메뉴의 재고 정보와 가격 정보 활용을 위해 음료수와 스낵의 정보는 다음의 파일을 이용합니다. 각 파일에는 메뉴별로 메뉴이름, 가격, 재고량 정보가 있으며 각 정보들은 \t으로 구분되어 있습니다.

- 음료수: drink2.txt
- 스낵: snack2.txt

우리는 재고 관리와 영수증 발급을 위해 2단계에서 구현하였던 프로그램을 확장합니다. 이때, 다음의 내용을 주의해서 구현하세요.


- 각 메뉴는 4가지 정보(메뉴번호, 메뉴이름, 가격, 재고량)가 있기 때문에 구조체로 구현합니다.
- 음료수와 스낵 리스트는 복수의 메뉴를 포함하기 때문에 구조체 배열로 구현합니다. (구조체 포인터 배열과 동적 메모리 할당 사용, 일반 배열로 구현 시 50% 감점)
- 음료수와 스낵 리스트를 포함하는 전체 메뉴 리스트는 구조체로 구현합니다.
- 사용자 주문 내역은 복수의 메뉴를 포함하기 때문에 구조체 배열로 구현합니다. (구조체 포인터 배열과 동적 메모리 할당 사용, 일반 배열로 구현 시 50% 감점)

각 함수들은 다음과 같은 기능을 수행하도록 확장 또는 신규로 구현하세요.

- main() 함수는 2단계 기능을 포함하며, 서비스 모드로 로그인 하였을 경우 sevice_mode()를 이용하여 주문 메뉴를 출력하고, 주문 결과를 알려줍니다.
- reset_drink_menu() 함수는 drink2.txt 파일에서 음료 메뉴 정보들을 읽어서 저장합니다.

- `reset_snack_menu()` 함수는 `snac2k.txt` 파일에서 스낵 메뉴 정보들을 읽어서 저장합니다.
- `service_mode()` 함수는 화면에 주문 메뉴를 출력하고, 주문을 받으며, 사용자가 주문을 종료하면 영수증을 저장하고, 화면에 출력합니다. 이를 위해 다음의 사항을 주의해서 구현하세요.
 - 주문 메뉴에는 각 메뉴의 메뉴번호, 메뉴이름, 가격 정보를 출력해야 합니다.
 - 사용자는 메뉴번호 입력을 통해 주문을 하며, 주문 메뉴는 사용자가 복수의 메뉴 주문이 가능하도록 구성되어야 합니다.
 - 재고량이 다 소진된 메뉴는 더 이상 주문을 받을 수 없습니다. 따라서, 사용자로부터 주문을 받으면 해당 메뉴의 재고량을 감소시키고, 재고량이 다 소진된 메뉴는 주문 메뉴에서 배제시켜야 합니다. 이를 위해 `reset_drink_menu()`, `reset_snack_menu()`를 이용해서 전체 메뉴 리스트를 만들고 관리하세요.
 - 사용자 주문 내역을 저장하고, `print_receipt()`를 이용해서 영수증을 발급합니다.
- `print_receipt()` 함수는 사용자가 주문한 메뉴의 총액을 계산해서 영수증을 발급합니다. 영수증은 `receipt.txt` 파일에 내용을 기록하고, 화면에도 출력합니다.

<주문 메뉴 예시>

<pre> ===== drink Menu ===== 1 : americano 1500 10 2 : caffe latte 2500 5 3 : ice tea 2000 7 4 : orange juice 2000 3 5 : mango juice 3000 4 6 : cold brew 3000 2 7 : Peppermint 3500 3 8 : Earl Gray 3500 11 9 : Chamomile 3500 7 11 : caffe moca 3000 5 12 : Caramel Macchiato 3500 3 13 : milk shake 4000 5 14 : lemonade 3500 1 ===== snack Menu ===== 1 : cream cake 4000 8 2 : cream bagel 3000 5 3 : scone 3000 3 4 : apple pie 3500 4 5 : cheese cake 5000 6 6 : tiramisu 5500 5 7 : egg tart 4000 3 </pre>	<pre> Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 14 Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 14 Out of stock! Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 6 Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 6 Type 1 for drink, 2 for snack, 3 for buy : 1 Type drink id to buy : 6 Out of stock! Type 1 for drink, 2 for snack, 3 for buy : 3 ===== Receipt ===== lemonade 3500 cold brew 3000 cold brew 3000 Total : 9500 ===== </pre>								
주문 메뉴 예시	<div style="text-align: center;">주문 리스트</div> <div>  receipt.txt - Windows 메모장 </div> <div> 파일(F) 편집(E) 서식(O) 보기(V) 도움말(H) </div> <table> <tr> <td>lemonade</td> <td>3500</td> </tr> <tr> <td>cold brew</td> <td>3000</td> </tr> <tr> <td>cold brew</td> <td>3000</td> </tr> <tr> <td>Total : 9500</td> <td></td> </tr> </table>	lemonade	3500	cold brew	3000	cold brew	3000	Total : 9500	
lemonade	3500								
cold brew	3000								
cold brew	3000								
Total : 9500									
	영수증 예시								

[4] 관리자 모드: 재고 주문 기능 개발

관리자 모드에서는 재고가 다 떨어진 메뉴의 경우 외부회사에 주문을 해서 재고를 채워 넣습니다. 4단계에서는 여러분의 프로그램 설계 역량을 확인하기 위해 간단한 기능 요구사항만을 제시합니다.

재고 주문은 다음의 알고리즘을 통해 이루어집니다.

- 시작 보유금은 30000
- 메뉴 판매 시 관련 금액만큼 보유금이 상승
- 재고 주문은 재고량이 0이 된 메뉴가 3개 이상 시 수행
- 재고 주문수는 5개 단위로 주문
- 주문 대상 외부회사 선택은 주문 메뉴의 재고가 모두 있으면서 금액이 가장 낮은 회사
- 재고 주문 시 주문 금액만큼 보유금이 감소 (금액이 부족할 시 주문 가능한 만큼만 주문)

재고 주문을 위해 다음의 파일을 이용합니다.

- 음료수: drink2.txt
- 스낵: snack2.txt
- 보유금 : asset.txt
- 외부회사: coupang.txt, 11st.txt, gmarket.txt

본 프로그램을 완성하기 위해 3단계에서 구현한 함수들을 수정하고, 추가로 재고 주문을 하는 함수를 신규로 구현합니다. 프로그램 전체에서는 동적 메모리 할당을 사용해야 합니다.

- admin_mode() 함수는 현재 재고량을 확인하고, 주문 대상 외부회사를 선택한 후 주문을 수행합니다. 주문 수행은 우리의 재고량 상승 및 보유금 감소를 통해 이루어집니다.
- order_print() 함수는 재고 주문 결과를 출력합니다. 출력 내용에는 재고 주문 리스트와 각 비용, 그리고 보유금 변동 정보가 포함되어야 합니다.


```

===== Sold out Menu =====
7 : Peppermint 3500 0
10 : green tea 3500 0
1 : cream cake 4000 0
4 : apple pie 3500 0

```

재고 없는 물품 출력

```

===== Order List =====
Peppermint 2800 5
green tea 1800 3
cream cake 2500 5
apple pie 2800 3
asset : 2200

```

구매 물품 및 예산 출력

```

===== Gmarget List =====
1 : americano 600 6
2 : caffe latte 1800 5
3 : ice tea 2000 8
4 : orange juice 1500 1
5 : mango juice 2200 9
6 : cold brew 3000 0
7 : Peppermint 2800 6
8 : Earl Gray 2200 9
9 : Chamomile 2000 5
10 : green tea 1800 3
11 : caffe moca 2300 2
12 : Caramel Macchiato 3100 1
13 : milk shake 3000 4
14 : lemonade 2800 8

```

각 회사별 물품 리스트 출력(1)

```

===== Coupang List =====
1 : americano 800 7
2 : cream cake 3300 2
3 : cream bagel 2000 3
4 : scone 2200 1
5 : caffe latte 2300 3
6 : apple pie 2800 3
7 : ice tea 1800 0
8 : orange juice 1600 5
9 : mango juice 2400 4
10 : cheese cake 4000 2
11 : tiramisu 5200 7
12 : egg tart 3800 8
13 : cold brew 2800 0
14 : Peppermint 3000 3
15 : Earl Gray 2200 4
16 : Chamomile 2500 6
17 : green tea 2000 7
18 : caffe moca 2300 7
19 : Caramel Macchiato 3200 8
20 : milk shake 2500 2
21 : lemonade 3000 3

```

```

===== 11st List =====
1 : cream cake 2500 8
2 : cream bagel 1500 5
3 : scone 2000 3
4 : apple pie 3000 4
5 : cheese cake 4300 6
6 : tiramisu 4400 0
7 : egg tart 3200 3

```

각 회사별 물품 리스트 출력(2)