제 16 강

카페프로그램 구현하기

교재: x

목차

카페프로그램

<실습 Cafe.java>

카페 프로그램 만들기!

1. 알고리즘 작성하기
2. 알고리즘을 토대로 코드 구현하기

MENU

1.	아메키라노	3800 원
2.	에스프레소	2400 원
3.	카페라떼	4200 원
4.	밀크티	5100 원

카페프로그램

1. 카페 프로그램 알고리즘 짜기

알고리즘 이란?

문제를 해결하기 위한 절차적인 과정

예시?

문제상황: 배가 고파서 라면이 먹고 싶다는 문제 발생!

문제해결 상황: 라면을 먹었다

카페프로그램

1. 카페 프로그램 알고리즘 짜기

1. 카페 프로그램에는 어떤 기능이 있는가?
2. 각 기능을 구현하기 위해선 어떤 작업을 해야 하는가?

1) 카페프로그램의 기능

(1) 주문하기
(2) 취소하기
(3) 결제하기
(4) 끝내기

1. 주문 가능한 메뉴를 출력한다.
2. 주문 받을 메뉴를 입력 받는다.
3. 주문한 메뉴의 가격을 총 금액에 누적한다.
4. 주문한 메뉴를 전체 메뉴리스트에 추가한다.
5. 주문한 개수를 한 개 증가한다.

카페프로그램

1. 카페 프로그램 알고리즘 짜기

1. 카페 프로그램에는 어떤 기능이 있는가?
2. 각 기능을 구현하기 위해선 어떤 작업을 해야 하는가?

1) 카페프로그램의 기능

(1) 주문하기
(2) 취소하기
(3) 결제하기
(4) 끝내기

1. 주문한 메뉴리스트를 출력한다.
2. 그 중에서 취소한 메뉴를 입력 받는다.
3. 입력 받은 메뉴를 주문리스트에서 제거한다.
4. 입력 받은 메뉴의 금액을 총 금액에서 차감한다.
5. 주문한 개수를 한 개 감소한다.

카페프로그램

1. 카페 프로그램 알고리즘 짜기

1. 카페 프로그램에는 어떤 기능이 있는가?
2. 각 기능을 구현하기 위해선 어떤 작업을 해야 하는가?

1) 카페프로그램의 기능

(1) 주문하기
(2) 취소하기
(3) 결제하기
(4) 끝내기

1. 결제해야 할 총 금액을 출력한다.
2. 사용자에게 지불할 금액을 입력 받는다.
3. 지불한 금액과 총 금액을 비교해서,

만약, 지불한 금액 < 총 금액이면?
" 잔돈이 부족합니다" 출력 후 결제 취소

그게 아니라면,
잔돈과 함께 계산완료를 출력한다.

4. 총 금액은 0원으로 주문리스트는 비워준다.
5. 전체 개수를 0개로 초기화한다.

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<사용할 개념>

1. 무한반복
2. 문자열 배열

무한반복?

: 반복이 종료 되지 않고 계속 실행되는 구조
따라서, 종료조건과 함께 사용!

문자열 배열?

: 배열의 각 요소가 문자열(String)인 배열

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<필요한 변수>

1. 전체 개수 저장할 변수
2. 주문한 메뉴를 저장할 문자열 배열
3. 총 금액 저장할 변수

```
int count=0;
String orderList[]=new String[5];
int total=0;
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<기본 틀 작성>

입력에 따라 해야 할 명령이 다르므로
조건문 사용!

```
while(true) {
    System.out.println("☆★☆★ Cafe ☆★☆★");
    System.out.println("1. 주문하기");
    System.out.println("2. 취소하기");
    System.out.println("3. 결제하기");
    System.out.println("4. 끝내기");
    int num=sc.nextInt();
    if(num==1) {
    }
    else if(num==2) {}
    else if(num==3) {}
    else if(num==4) {
        System.out.println("프로그램 종료!");
         break;
    }
    else {
        System.out.println("잘못 입력하셨습니다.");
    }
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 주문하기

1. 주문 가능한 메뉴를 출력한다.
2. 주문 받을 메뉴를 입력 받는다.
3. 주문한 메뉴의 가격을 총 금액에 누적한다.
4. 주문한 메뉴를 전체 메뉴리스트에 추가한다.
5. 주문한 개수를 한 개 증가한다.

MENU☕

1.	아메키라노	3800 원
2.	에스프레소	2400 원
3.	카페라떼	4200 원
4.	밀크티	5100 원

```
if(num==1) {
    System.out.println("☆★☆★MENU☆★☆★");
    System.out.println("1.아메리카노\t3800 원");
    System.out.println("2.에스프레소\t2400 원");
    System.out.println("3.카페라떼\t4200 원");
    System.out.println("4.밀크티\t\t5100 원");
    System.out.print("주문할 메뉴:");
    int choice=sc.nextInt();

}
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 주문하기

1. 주문 가능한 메뉴를 출력한다.
2. 주문 받을 메뉴를 입력 받는다.
3. 주문한 메뉴의 가격을 총 금액에 누적한다.
4. 주문한 메뉴를 전체 메뉴리스트에 추가한다.
5. 주문한 개수를 한 개 증가한다.

MENU

1.	아메키라노	3800 원
2.	에스프레소	2400 원
3.	카페라떼	4200 원
4.	밀크티	5100 원

```
if(choice==1) {
    menuName="아메리카노";
    menuPrice=3800;
}
else if(choice==2) {
    menuName="에스프레소";
    menuPrice=2400;
}
else if(choice==3) {
    menuName="카페라떼";
    menuPrice=4200;}
else if(choice==4) {
    menuName="밀크티";
    menuPrice=5100;
}
else {
    System.out.println("잘못입력하셨습니다.");
    continue;
}
System.out.println("주문한 메뉴는 "+menuName+"입니다.");
System.out.println("가격은 "+menuPrice+"원 입니다.");
orderList[count]=menuName;
total+=menuPrice;
count++;
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 취소하기

1. 주문한 메뉴리스트를 출력한다.
2. 그 중에서 취소한 메뉴를 입력 받는다.
3. 입력 받은 메뉴를 주문리스트에서 제거한다.
4. 입력 받은 메뉴의 금액을 총 금액에서 차감한다.
5. 주문한 개수를 한 개 감소한다.

```
else if(num==2) {
    for(int i=0;i<count;i++) {
    System.out.println(i+1+":"+orderList[i]);
    }
    System.out.print("취소할 메뉴 번호:");
    int cancel=sc.nextInt();
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 취소하기

1. 주문한 메뉴리스트를 출력한다.
2. 그 중에서 취소한 메뉴를 입력 받는다.
3. 입력 받은 메뉴를 주문리스트에서 제거한다.
4. 입력 받은 메뉴의 금액을 총 금액에서 차감한다.
5. 주문한 개수를 한 개 감소한다.

```
if(1<=cancel && cancel<=count) {
    String delMenu=orderList[cancel-1];
    System.out.println(delMenu+" 메뉴 삭제!");
    for(int i=cancel-1;i<count;i++) {
        orderList[i]=orderList[i+1];
    }
```

orderList

아메리카노	에스프레소	카페라떼	밀크티	
[0]	[1]	[2]	[3]	[4]

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 취소하기

1. 주문한 메뉴리스트를 출력한다.
2. 그 중에서 취소한 메뉴를 입력 받는다.
3. 입력 받은 메뉴를 주문리스트에서 제거한다.
4. 입력 받은 메뉴의 금액을 총 금액에서 차감한다.
5. 주문한 개수를 한 개 감소한다.

```
        if(delMenu.equals("아메리카노")){
            total-=3800;
        }
        else if(delMenu.equals("에스프레소")){
            total-=2400;
        }
        else if(delMenu.equals("카페라떼")){
            total-=4200;
        }
        else if(delMenu.equals("밀크티")){
            total-=5100;
        }
        count--;
    }
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

출력형태 변경하기

1. 현재 금액 출력하기
2. 주문한 내역이 있을 경우 출력하기

```
System.out.println("☆★☆★ Cafe ☆★☆★");
System.out.println("현재 금액:"+total+"원");
System.out.println("주문내역-------");
for(int i=0;i<count;i++) {
    System.out.println(i+1+". "+orderList[i]);
}
```

```
☆★☆★ Cafe ☆★☆★
현재 금액:0원
주문내역-------
---------------
1. 주문하기
2. 취소하기
3. 결제하기
4. 끝내기
입력:
```

[첫 화면]

```
☆★☆★ Cafe ☆★☆★
현재 금액:6200원
주문내역-------
1. 아메리카노
2. 에스프레소
---------------
1. 주문하기
2. 취소하기
3. 결제하기
4. 끝내기
입력:
```

[주문 후 화면]

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 결제하기

1. 결제해야 할 총 금액을 출력한다.
2. 사용자에게 지불할 금액을 입력 받는다.
3. 지불한 금액과 총 금액을 비교해서,
만약, 지불한 금액 < 총 금액이면?
" 잔돈이 부족합니다" 출력 후 결제 취소
그게 아니라면,
잔돈과 함께 계산완료를 출력한다.
4. 총 금액은 0원으로 주문리스트는 비워준다.
5. 전체 개수를 0개로 초기화한다.

```
else if(num==3) {
    System.out.println("결제해야할 금액: "+total+"원");
    System.out.print("지불할 금액:");
    int money=sc.nextInt();
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 결제하기

1. 결제해야 할 총 금액을 출력한다.
2. 사용자에게 지불할 금액을 입력 받는다.
3. 지불한 금액과 총 금액을 비교해서,
만약, 지불한 금액 < 총 금액이면?
" 잔돈이 부족합니다" 출력 후 결제 취소
그게 아니라면,
잔돈과 함께 계산완료를 출력한다.
4. 총 금액은 0원으로 주문리스트는 비워준다.
5. 전체 개수를 0개로 초기화한다.

```
if(money<total) {
    System.out.println("잔돈이 부족합니다.");
    continue;
}
else{
    System.out.println("잔돈은 "+(money-total)+"원 입니다.");
    System.out.println("결제가 완료되었습니다.");
}
```

카페프로그램

2. 알고리즘을 토대로 코드 구현하기

<상세 기능에 따른 코드 구현>- 결제하기

1. 결제해야 할 총 금액을 출력한다.
2. 사용자에게 지불할 금액을 입력 받는다.
3. 지불한 금액과 총 금액을 비교해서,
만약, 지불한 금액 < 총 금액이면?
" 잔돈이 부족합니다" 출력 후 결제 취소
그게 아니라면,
잔돈과 함께 계산완료를 출력한다.
4. 총 금액은 0원으로 주문리스트는 비워준다.
5. 전체 개수를 0개로 초기화한다.

```
total=0;
for(int i=0;i<orderList.length;i++) {
    orderList[i]="";
}
count=0;
```