실습자료

여러분은 “ GoCafe” 를 운영하는 카페 사장입니다 .
실습 전체에 걸쳐 카페를 운영하는 데에 필요한 프로그램을 만들 것입니다 .

실습 1 변수 선언하기

카페 이름 , 아메리카노 가격 , 라떼 가격 , 카페 전화번호 , 오늘 여는지 여부를 선언해봅시다 . 그리고 모두 출력해봅시다 . 각각 어떤 자료형이 필요한지 생각해보세요 !

카페이름 - 변수이름 : cafeName 값 : GoCafe
아메리카노 가격 - 변수이름 : americanoPrice 값 : 1500
라떼 가격 - 변수이름 : lattePrice 값 : 1700
카페 전화번호 - 변수이름 : cafePhoneNumber 값 : 02-428-5544
오늘 여는지 여부 - 변수이름 : isOpenToday 값 : true

실습 2 배열 사용하기

GoCafe 에는 메뉴가 아메리카노 , 라떼 두 개 있습니다 . 이 두 메뉴의 이름이 저장된 배열 (drinkNames) 과 가격이 저장된 배열 (drinkPrices) 을 정의해주세요 .

실습 3 연산자 사용하기

어떤 손님이 아메리카노 M 잔과 라떼 N 잔을 주문하였습니다 . 총 가격을 출력해주세요 .
오늘은 개업날이니 10% 할인을 해줍시다 . 할인된 가격을 출력해주세요 .
(Hint! 정수와 소수를 계산하고 싶을 때에는 명시적형변환을 해주어야 합니다 .)

실습 4 조건문 사용하기

GoCafe 는 할인을 많이 해주는 카페입니다 .
1) 총 금액이 5000 원 초과일 경우 10% 를 할인해줍시다 .
2) 총 금액이 5000 원 초과이면서 , 월요일인 경우엔 추가로 10% 를 또 할인해줍니다 .
3) 같은 음료를 4 잔 이상 주문시 10% 할인
한 손님이 아메리카노와 라떼를 주문하였습니다 . 아래를 참고하여 총 금액을 계산해보세요 .

```
1  package main
2  import "fmt"
3  func main(){
4    isTodayMonday := true
5    drinkNames := [2]string{"Americano", "Latte"}
6    drinkPrices := [2]int{1500, 1700}
7
```

실습 5 반복문 사용하기

GoCafe 에 메뉴가 5 개로 늘었습니다 .
어떤 VIP 손님이 메뉴에 있는 음료를 처음부터 끝까지 주문했습니다 .
for 문을 사용하여 총 금액을 계산해봅시다 .

```
5  func main(){
6      drinkNames  := []string{"Americano", "Latte", "CafeMocha", "GreenTeaLatte", "HotChoco"}
7      drinkPrices := []int{1500, 1700, 1900, 2300, 2000}
8
9      totalPrice := 0
```

실습 5 예시답안

```go
func main(){
    drinkNames  := []string{"Americano", "Latte", "CafeMocha", "GreenTeaLatte", "HotChoco"}
    drinkPrices := []int{1500, 1700, 1900, 2300, 2000}

    totalPrice := 0

    for _, i := range drinkPrices {
        totalPrice += i
    }
    fmt.Println("총 금액은", totalPrice, "입니다.")
}
```

실습 6 반복문 내에서 조건문 사용하기

메뉴가 또 추가되었습니다 . 손님은 , 2000 원 이하인 음료만 주문하고 싶어합니다 .
또한 총 금액이 7000 원 이상 되면 더 이상 주문하지 않는다고 합니다 .
주문한 메뉴와 , 최종 금액을 출력해주세요 .
Hint) break 와 continue 를 적절히 활용해보세요

```
5  func main(){
6    drinkNames  := []string{"Americano", "Latte", "CafeMocha", "GreenTeaLatte",
7      "HotChoco", "ChamomileTea", "StrawberrySmoothie","HotMilk"}
8    drinkPrices := []int{1500, 1700, 1900, 2300, 2000, 1800, 3500, 1500}
9
10   totalPrice := 0
11   orderedDrinks := ""
```

실습 7 함수 사용하기

카페에 전화를 건 손님에게 인사한 후 , 요일에 따라 마감시간을 안내하는 함수를 만들어 봅시다 .
월요일 마감시간 : 10 시
화요일 마감시간 : 9 시
수요일 마감시간 : 10 시
목요일 마감시간 : 9 시
금요일 마감시간 : 10 시
토요일 마감시간 : 8 시
일요일 : 휴무

```
19  func main(){
20          greeting()
21          greeting2("수요일")
22  }
```

실습 8 구조체 정의하기

메뉴가 하나 늘어나면 이름 배열 , 가격 배열을 수정해주어야 합니다 . 그런데 손님들이 음료에 카페인이 있는지 없는지도 표기해달라고 합니다 . 그러면 또 카페인함유여부 배열을 만들고 일일이 추가해주어야 하겠죠 ? 그러한 번거로움을 줄이기 위해 Menu 라는 구조체를 만들어 봅시다 .

Menu : 메뉴이름 (name), 가격 (price), 카페인함유여부 (isCoffee)

실습 9 구조체 정의하기

GoCafe 는 스타벅스처럼 멤버십이 있습니다 . 멤버십에 가입하는 손님의 이름 , 나이 , 핸드폰번호 , VIP 여부 , 적립된 포인트가 포함된 Customer 구조체를 만들어봅시다 .

Customer : 이름 (name), 나이 (age), 핸드폰번호 (phoneNumber), 단골여부 (isVip), 포인트 (point)

실습 8 예시답안

```
4  type Menu struct {
5          name string
6          price int
7          isCoffee bool
8  }
```

실습 9 예시답안

```
4  type Customer struct {
5          name string
6          age int
7          phoneNumber string
8          isVip bool
9          point int
10 }
```

실습 10 통합 예제

이제 메뉴와 멤버십 배열을 만들어 봅시다 .
이 구조체 배열들을 이용하여 여러 가지 함수를 만들어봅시다 .

```
4   type Customer struct {
5           name string
6           age int
7           phoneNumber string
8           isVip bool
9           point int
10  }
11
12  type Menu struct {
13          name string
14          price int
15          isCoffee bool
16  }
17
18  var menus       [5]Menu
19  var membership  [5]Customer
```

```
21  func main(){
22    menus[0] = Menu{"Americano"     , 1500  , true}
23    menus[1] = Menu{"Latte"         , 1700  , true}
24    menus[2] = Menu{"CafeMocha"     , 1900  , true}
25    menus[3] = Menu{"GreenTeaLatte" , 2300  , false}
26    menus[4] = Menu{"HotChoco"      , 2000  , false}
27
34
35    membership[0] = Customer{"Tom", 23, "010-3212-5578", true   , 3200}
36    membership[1] = Customer{"Sam", 17, "010-987-5959" , true   , 2000}
37    membership[2] = Customer{"Kim", 42, "010-529-1982" , false  , 1000}
38
39  }
```

실습 10 통합 예제

1) 전체 메뉴 중에서 2000 원 미만인 음료만 출력해봅시다 . (함수명 : printLowPrice)
2) 전체 메뉴 중에서 카페인이 함유된 음료만 출력해봅시다 . (함수명 : printCoffee)
3) 포인트가 2000 점 이상이 되면 VIP 가 됩니다 . VIP 는 10% 할인을 해 줍니다 . 포인트는 주문 금액의 10% 가 적립됩니다 .
4) 주문을 받는 함수 Order 을 만듭시다 . 이 함수 안에는 포인트를 적립하는 함수 , VIP 여부를 확인하고 , 포인트에 따라 VIP 여부를 바꾸어주는 함수가 포함됩니다 .

실습 10 예시답안

1)

```
func printLowPrice() {
    fmt.Println("2000원 미만 메뉴는 다음과 같습니다")
    for _, menu := range menus {
        if menu.price < 2000 {
            fmt.Println(menu.name)
        }
    }
}
```

2)

```
func printCoffee() {
    fmt.Println("커피 메뉴는 다음과 같습니다")
    for _, menu := range menus {
        if menu.isCoffee == true {
            fmt.Println(menu.name)
        }
    }
}
```

실습 10 예시답안

3,
4)

```
func main(){
  menus[0] = Menu{"Americano"     , 1500  , true}
  menus[1] = Menu{"Latte"         , 1700  , true}
  menus[2] = Menu{"CafeMocha"     , 1900  , true}
  menus[3] = Menu{"GreenTeaLatte" , 2300  , false}
  menus[4] = Menu{"HotChoco"      , 2000  , false}

  membership[0] = Customer{"Tom", 23, "010-3212-5578", true   , 3200}
  membership[1] = Customer{"Sam", 17, "010-987-5959" , true   , 2000}
  membership[2] = Customer{"Kim", 42, "010-529-1982" , false  , 1000}

  fmt.Println("가격은 ", order(2, 0), "원 입니다.") //TOM이 카페모카 주문
  fmt.Println(membership[0].name, "님의 잔여 포인트는 ", membership[0].point, "입니다." )

}
```

실습 10 예시답안

3,
4)

```
func order(menuNum int, customerNum int) int {
  totalPrice := 0
  //VIP인경우 10% 할인
  if(membership[customerNum].isVip){
    totalPrice = int(float32(menus[menuNum].price) * 0.9)
  }else{
    totalPrice = menus[menuNum].price
  }
  //포인트 적립
  addPoints(totalPrice, customerNum)
  return totalPrice
}

func addPoints(price int, customerNum int){
  membership[customerNum].point += int(float32(price) * 0.1)
  //포인트가 2000점 이상일 경우 VIP가 됩니다.
  if(membership[customerNum].point >= 2000){
    membership[customerNum].isVip = true
  }
}
```