Q1. 아래 코드를 모든 예외를 처리할 수 있도록 보완하세요

In [87]:

```
list_num = [52,273,32,72,100]
try:
    num_input = int(input('정수 입력> '))
    print('{}번째 요소: {}'.format(num_input, list_num[num_input]))
    예외.발생()
except ValueError as exception:
    print('정수를 입력해 주세요.')
    print(type(exception),exception)
except IndexError as exception:
    print('리스트의 인덱스를 벗어났어요.')
    print(type(exception),exception)
finally:
    print('모든 예외 처리')
```

. . .

In [93]:

```
list_num = [52,273,32,72,100]
try:
    num_input = int(input('정수 입력> '))
    print('{}번째 요소: {}'.format(num_input, list_num[num_input]))
    예외.발생()
except ValueError as exception:
    print('정수를 입력해 주세요.')
    print(type(exception),exception)
except IndexError as exception:
    print('리스트의 인덱스를 벗어났어요.')
    print(type(exception),exception)
except NameError as exception:
    print("'예외.발생()' 값이 지정되지 않았습니다.")
    print(type(exception),exception)
finally:
    print('모든 예외 처리')
```

```
정수 입력> 1
1번째 요소: 273
'예외.발생()' 값이 지정되지 않았습니다.
<class 'NameError'> name '예외' is not defined
모든 예외 처리
```

Q2. 기본가격 1000원인 3개의 상품에 대하여 임의의 추가 가격을 인수로 대입시 더한 가격을 산출하세요 (클래스이용)

In [3]:

```
add_price = int(input("추가값을 입력하세요"))
class Price:
    def get_3_price(self):
        item1 = 1000
        item2 = 1000
        item3 = 1000
        global add_price
        item1 += add_price
        item2 += add_price
        item3 += add_price
        return print(item1,item2,item3)

pri=Price()
pri.get_3_price()
```

```
추가값을 입력하세요1000
2000 2000 2000
```

Q3. 기본가격 1000원인 2개의 상품에 대하여 임의의 추가 가격을 입력시 아래 두개의 방식으로 산출하세요(class이용)

- price 1 : 기본가격 + 추가가격
- price 2 : (기본가격 + 추가가격) * 90%

In [4]:

```
op1 = 1000
op2 = 1000
price = int(input("추가 가격을 입력하세요:"))
class Getprice:
    def Add_price(self):
        global op1
        op1 += price
        return op1

    def Discount_price(self):
        global op2
        result_Discount_price = int((op2 + price) * 0.9)
        return result_Discount_price

Answer1 = Getprice()
print(Answer1.Add_price())
Answer2 = Getprice()
print(Answer2.Discount_price())
```

```
추가 가격을 입력하세요:1000
2000
1800
```

Q4. 임의의 클래스를 작성한 후 인스턴스를 생성하고 그것의 타입을 확인하세요

In [5]:

```
class Class:
    def __init__(self,name):
        self.name = name

    def Who(self):
        print(self.name)

    def with_whom(self):
        print("same with %s" %self.name)

A3 = Class("Dum")
A3.Who()
type(A3)
```

```
Dum
```

```
__main__.Class
```

Q5 4칙 연산 기능을 포함한 Cal4 클래스(생성자이용)를 작성하고 이 클래스를 이용하여 cal1계산기 객체를 만든 후 두개의 수 5,3에 대한 사칙연산을 수행하세요

In [6]:

```
class Cal4:
    def __init__(self,a,b):
        self.a = a
        self.b = b

    def sum(self):
        result_sum = self.a + self.b
        return result_sum

    def sub(self):
        result_sub = self.a - self.b
        return result_sub

    def mul(self):
        result_mul =  self.a * self.b
        return result_mul

    def div(self):
        if self.b == 0:
            print("Cannot divide 0")
        else:
            result_div =  self.a / self.b
            return result_div
cal1 = Cal4(5,0)
print(cal1.sum())
print(cal1.sub())
print(cal1.mul())
print(cal1.div())
```

```
5
5
0
Cannot divide 0
None
```

Q6. Order 클래스를 상속받아 extraorder 클래스를 작성하고 extraCustomer의 주문가격을 다음과 같이 산출하였다. ㄷ extraOrder클래스를 작성하세요

In [7]:

```
class Order:
    def __init__(self,name):
        self.customer = 0
        self.name = name
    def order(self,price):
        self.customer += price
        return self.customer

class ExtraOrder(Order):
    pass

extraCustomer =ExtraOrder('Kevin')
print(extraCustomer.order(1000))
```

```
1000
```

Q7. Order 클래스를 상속받아 extraOrder 클래스에서 메소드 오버라이딩하여 출력가격에 '원'이 추가되도록 출력하세요

In [8]:

```
# method를 '원'이 추가되도록
class Order:
    def __init__(self,name):
        self.customer = 0
        self.name = name
    def order(self,price):
        self.customer += price
        return self.customer

class ExtraOrder(Order):
    def order(self,price):
        self.customer += price
        return print("가격은 {}{}입니다.".format(self.customer,"원"))

extraCustomer =ExtraOrder('Kevin')
print(extraCustomer.order(1000))
```

```
가격은 1000원입니다.
None
```

Q8. 업무미팅이 2시임을 알려주는 자동 이메일을 클래스 AutoEmail을 작성하여 아래와 같이 출력하세요.

- 안녕하세요 Kevin님
- 업무미팅은 2시 입니다.

In [9]:

```
class AutoEmail:
    def __init__(self,name):
        self.name = name
        return print( "안녕하세요,%s님" %name)
    def Meeting(self):
        import datetime as dt
        if dt.datetime.hour == 2:
            print("업무미팅이 {}시에 있습니다.".format(dt.datetetime.hour))
        else:
            print("업무미팅시간이 아닙니다.")
            print("현재시각은 {}입니다.".format(dt.datetime.now()))
email = AutoEmail('Kevin')
email.Meeting()
```

```
안녕하세요,Kevin님
업무미팅시간이 아닙니다.
현재시각은 2021-06-28 17:20:46.969685입니다.
```

Q9. 1609160537.371015를 "Sat Jun 26 08:35:03 2021" 포맷으로 출력하세요

In [10]:

```
import time
time.strftime('%c',time.localtime(1609160537.371015))
```

```
'Mon Dec 28 22:02:17 2020'
```

Q10. 현재 날짜와 시간을 "Sat Jun 26 08:35:03 2021" 포맷으로 출력하세요

In [11]:

```
import time
nowtime = time.time()
print(nowtime)
clean_nowtime = time.strftime('%c',time.localtime(time.time()))
print(clean_nowtime)
```

```
1624868451.6040964
Mon Jun 28 17:20:51 2021
```

Q11. 현재 시간을 년-월-일 시:분:초로 출력하세요

In [12]:

```
import datetime as dt
now = dt.datetime.now()
print(now)
```

```
2021-06-28 17:20:52.665045
```

Q12. 올해 경과된 날짜수 계산하세요

In [13]:

```
from datetime import datetime as dt
now = dt.now()
print(now)

now_compare = dt.strptime("20210101","%Y%m%d")
print(now_compare)

compare_date_difference = now - now_compare
print(compare_date_difference)

compare_date_difference.days
```

```
2021-06-28 17:20:53.575435
2021-01-01 00:00:00
178 days, 17:20:53.575435



178
```

Q13. 현재 요일을 "2021-6-26 오늘은 토요일입니다."와 같은 형식으로 출력하세요

In [15]:

```
import datetime as dt
def print_time_now_english():
    t =dt.datetime.now()
    print(t.strftime("%Y-%m-%d, Today: %A "))

print_time_now_english()
```

```
2021-06-28, Today: Monday
```

In [21]:

```
import datetime as dt
import locale
locale.setlocale(locale.LC_ALL, 'ko_KR.UTF-8')
def print_time_now_korean():
    t = dt.datetime.now()
    print(t.strftime("%Y-%m-%d, 오늘은 %A 입니다"))


print_time_now_korean()
```

```
2021-06-28, 오늘은 월요일 입니다
```

Q14. 1에서 백만까지 더하는데 걸리는 프로그램 실행 시간을 밀리초(ms) 단위로 구하세요

- 1000 밀리초(ms) = 1초.

In [42]:

```
import timeit
time_start = timeit.default_timer()
for i in range(1,1000001):
    i += 1
time_stop = timeit.default_timer()
print(time_stop-time_start)
```

```
0.10676920000003065
```

In [47]:

```
import time
starttime = time.time()
for i in range(1,1000001):
    i += 1
stoptime = time.time()
program_time = stoptime-starttime
print(program_time)
```

```
0.10807156562805176
```

Q15. Bonus

name = ['고영남', '김광훈', '김동일', '김진', '박기범', '박민아', '박시우', '배송이', '송유빈', '신인철', '양인석','오수문','우동주','이덕재','이민찬','이범준','이슬','이원진','이종현','임희진', '정하림','조경림','조현정','진유훈','채승혜','최윤진','최한결','최혜정','하도원','안아름']

- 발표할 인원을 입력하면 랜덤으로 발표자를 리스트로 출력해주는 프로그램을 작성하세요.(중복 허용하지 않음)

In [62]:

```
import random
name = ['고영남', '김광훈', '김동일', '김진', '박기범',
        '박민아', '박시우', '배송이', '송유빈', '신인철',
        '양인석','오수문','우동주','이덕재','이민찬','이범준',
        '이슬','이원진','이종현','임희진', '정하림','조경림','조현정',
        '진유훈','채승혜','최윤진','최한결','최혜정','하도원','안아름']
#random_pick = random.sample(name,2)
#print(random_pick)
#type(random_pick)
```

In [85]:

```
name = ['고영남', '김광훈', '김동일', '김진', '박기범',
        '박민아', '박시우', '배송이', '송유빈', '신인철',
        '양인석','오수문','우동주','이덕재','이민찬','이범준',
        '이슬','이원진','이종현','임희진', '정하림','조경림','조현정',
        '진유훈','채승혜','최윤진','최한결','최혜정','하도원','안아름']
def today_presenters():
    global name
    name_clean= list(set(name))
    already = []
    counts = int(input("오늘의 발표자는 몇 명? : "))
    if counts > len(name_clean):
        print("학생보다 많습니다. 다시 입력해주세요:")
        return today_presenters()
    Shuffle = input("섞을까요? Y/N : ")
    if Shuffle == "Y":
        random.shuffle(name_clean)
    else:
        pass
    choiced_presenter = random.sample(name_clean, counts)
    return print(choiced_presenter)
```

In [86]:

```
today_presenters()
```

```
오늘의 발표자는 몇 명? : 2
섞을까요? Y/N : Y
['하도원', '배송이']
```

In [ ]: