<공지>

1. 각 문제를 해결하기 위한 함수가 포함된 파이썬 모듈 파일(.py)을 만듭니다.

a. 파일 이름은 "S\_20210018.py"이어야 합니다.

b. 문제 1과 문제 2의 함수 이름은 각각 "p1"과 "p2"여야 합니다.

c. 두 가지 문제가 있으므로 Python 모듈 파일에는 두 가지 문제에 대한 두 가지 함수("p1" 및 "p2")가 포함되어야 합니다.

2. 솔루션이 포함된 이 docx 파일을 완성합니다.

3. 마감일까지 (1) 모듈 파일과 (2) docx 파일을 제출합니다.

4. 코드가 제대로 작동하는지 확인할 때 파일 이름("S\_20210018")을 제외한 다른 변경 사항 없이 validation code를 사용합니다.

a. 일부 외부 모듈이 필요한 경우 validation code가 아닌 모듈 파일 내에서 가져와야 합니다.

**Problem 1: A traveling sales problem for 100 cities**

한 판매원이 100개의 도시를 여행하고 싶어합니다. 각 도시를 정확히 한 번 방문하고 출발 도시로 돌아오는 가장 짧은 경로는 무엇입니까? 당신이 얻을 수 있는 최선의 해결책을 찾으세요. 당신의 솔루션이 우수할수록 이 문제에서 더 높은 점수를 받게 됩니다.

이 문제를 해결하려면 csv 파일("Input\_data.csv")에서 도시에 대한 정보를 얻을 수 있습니다. csv 파일에서 origin city의 ID는 0입니다. a와 b두 도시 사이의 거리 D를 계산하기 위해, the Euclidean distance는 다음과 같이 적용한다.



텍스트이(가) 표시된 사진

자동 생성된 설명

이 TSP 문제에 대한 함수("p1")를 만듭니다.

이 기능은 모듈 파일("A\_123456.py ")에 포함되어야 합니다.

함수 이름("p1")을 변경하지 마십시오.

원하는 경우 다른 함수를 만들 수 있습니다. 이 경우 메인 함수("p1")는 다른 함수를 호출할 것입니다.

다음 validation code는 함수 grade를 매기는 데 사용됩니다.

라인 2: 유효성 검사 코드에서는 모듈 이름이 " S\_20210018"로 바뀝니다.

라인 4: 코드는 문제 1에 대한 csv 파일("Input\_data.csv")을 가져옵니다.

라인 5: 기능을 실행한 후 가능한 경로와 거리 값이 반환됩니다.

라인 7: 경로인 솔루션을 출력합니다.

해결책은 list이어야 합니다.

list의 요소는 정수여야 합니다.

라인 9: a real number(float)이어야 하는 솔루션의 총 거리를 출력합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

**Solution**

• 총 거리 값에는 마지막 도시와 출발 도시 사이의 반환 거리(returning distance)가 포함되어야 합니다.

• 경로는 0에서 시작하여 0으로 끝나야 합니다.

• 경로는 쉼표(,)로 나누어진 101개 도시(0, ..., 0)의 시퀀스여야 합니다.

예를 들어, 4개의 도시(0, 1, 2, 3)를 고려할 경우 0, 1, 2, 3, 0으로 경로를 나타낼 수 있습니다.

• 당신의 거리 값이 맞는지 다시 한번 확인해서 당신의 경로를 확인할 것입니다. 그것이 틀리면 당신은 이 문제를 실패하게 될 것입니다. 검증 시, 채점자는 표에 경로를 "복사하여 붙여넣기"합니다. 그러니 상자에 경로를 문자로 적어주세요. 그림으로 표현하지 마십시오.

|  |  |
| --- | --- |
| **Total distance of your route:** |  |
| **Route:** | *PUT* ***A TEXT*** *BUT* ***NOT A PICTURE*** *HERE* |

**References**

* If you want to cite some references, you can put them in the following table.

|  |
| --- |
|  |

**Problem 2: Facility location problem**

한 대기업이 자사 판매 센터로 제품을 배달하기 위해 새로운 창고를 열려고 합니다. 회사는 가능한 일련의 후보 창고 위치 중에서 어느 위치에 창고를 건설할지 결정해야 합니다. 만약 회사가 장소들 중 하나에 특정 창고를 짓기로 결정한다면, 그것들은 고정된 건설 비용으로 발생하게 될 것입니다. 또한, 입지의 크기 때문에, 만약 그것들이 건설된다면, 각각의 후보 기지는 특정한 수용력을 가질 것입니다.

상품들은 건설된 창고에서 판매 센터로 배달되며, 모든 배달은 상품 배송량에 따라 운송비가 다릅니다. 건설비용은 일반적으로 큰 자본비용인 반면 운송비용은 운영비용입니다.

신규 창고를 건설할 수 있는 입지는 5곳이며, 이들 창고를 통해 납품을 받아야 하는 영업소는 12곳이다. 모든 후보 창고에 대해 창고를 구축하는 데 드는 고정 비용과 용량 제한 등의 정보를 제공합니다. 그 정보는 다음 표에 제시되어 있다.

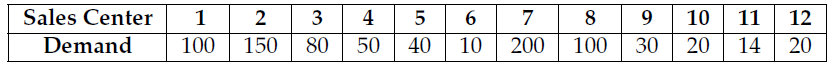
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Candidate depot | 1 | 2 | 3 | 4 | 5 |
| Construction cost | 1000 | 500 | 1500 | 300 | 800 |
| Capacity | 200 | 105 | 400 | 50 | 500 |

또한, 다음 표는 각 후보 창고에서 각 판매 센터로 제품 단위 배송 비용을 제공합니다. 판매 센터의 수요를 충족시키기 위해 제품은 하나 이상의 창고에서 배송될 수 있습니다.

테이블이(가) 표시된 사진

자동 생성된 설명

마지막으로 영업소에서 요구하는 수량을 다음 표에 정리했습니다. 당신의 임무는 당신의 모델이 각 판매 센터의 수요를 충족시키는 것을 보증하는 방식으로 이 문제를 공식화하는 것입니다. 모든 수요와 창고 용량을 충족시키면서 건설 및 배송의 총 비용을 최소화하기 위해 어떤 창고를 열어야 하는가?



• 설비 위치 문제에 대한 함수("p2")를 작성합니다.

o 이 기능은 모듈 파일("S\_20210018.py")에 포함되어야 합니다.

o 함수 이름("p2")을 변경하지 마십시오.

o 원하는 경우 다른 함수를 만들 수 있습니다. 이 경우 메인 함수("p2")는 다른 함수를 호출합니다.

• 다음 validation code는 함수 grade를 매기는 데 사용됩니다.

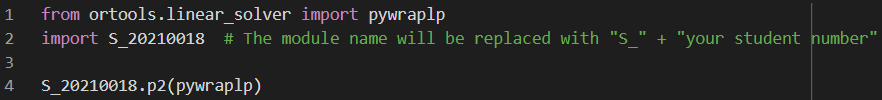
o 라인 1: OR-Tools 가져오기

o 라인 2: validation code에서 모듈 이름은 " S\_20210018"로 바뀝니다.

o 라인 4: 인수 pywraplp를 전달하여 주 함수("p2")를 호출합니다.

 솔루션의 표시 코드가 모듈 파일에 포함되어야 합니다.

 이 코드를 실행한 후 최적화의 결과로 최적의 솔루션과 목표값(총 비용)을 제시해야 합니다.

 **Solution**

|  |  |
| --- | --- |
| **Objective value**  **(total cost):** |  |
| **Optimal solution:** |  |

**References**

* If you want to cite some references, you can put them in the following table.

|  |
| --- |
|  |