Wstęp

Cześć… chciałbym Ci opowiedzieć o wszystkim, co robimy w Ocado, ale niestety nie mamy teraz na to czasu. Mamy problem: wybudowaliśmy ten wspaniały, zrobotyzowany magazyn, lecz nie możemy go uruchomić! Software, który miał planować trasy botów, nie został dostarczony na czas. Na jutro zaplanowane są testy magazynu, a nic nie działa. Jeśli się nam nie uda, możemy opóźnić cały go-live!

Mamy tylko 24 godziny na dostarczenie działającego programu! Czy nam w tym pomożesz?

Opis modelu magazynu

To, co widzisz na powyższym zdjęciu, to wnętrze naszego magazynu. Jest to grid, po którym jeżdżą nasze roboty. Każdy z magazynów Ocado składa się z kilku podstawowych elementów:

- Grid - górna powierzchnia przestrzeni, gdzie przechowujemy towar. Mogą po nim jeździć boty. Składa się z modułów i zawsze ma kształt prostokąta o wymiarach $0 < X \leq 100$ na $0 < Y \leq 100$. Modelujemy go, używając układu kartezjańskiego z odwróconą osią Y - patrz **ilustracja na końcu dokumentu**.
- Moduł magazynu - pojedyncza pionowa kolumna zbudowana z aluminiowych profili, po której robot może przejechać, ale też zaparkować i wyciągnąć jeden z N kontenerów ułożonych w głąb, jeden na drugim ($0 < N < 10$). Warstwy ponumerowane są od 0 do N-1 ($0 \leq n < N$), gdzie warstwa 0 to ta najbliżej Gridu. N to liczba warstw w danym magazynie. Nasze roboty mogą wyciągnąć kontener z każdej warstwy, ale im wyżej on jest (bliżej gridu), tym wyciągnie go szybciej.
- Stacja odbiorcza - specjalny moduł, gdzie boty mogą odkładać kontenery, aby później można było wyciągnąć z niego przedmiot i zapakować go do zamówienia klienta. Jeden wybrany brzegowy moduł magazynu nie może przechowywać towaru, jest natomiast stacją odbiorczą. Nie ma on warstw.
- Kontener - plastikowy pojemnik, który zawiera wybrany produkt (tylko jeden). Każdy z kontenerów identyfikujemy przez trójkę (x, y, n), określającej współrzędne na gridzie oraz warstwę w module, gdzie $0 \leq x < X$, $0 \leq y < Y$ oraz $0 \leq n < N$. W całym magazynie możemy przechowywać maksymalnie $X * Y * N$ kontenerów (minus stacja odbiorcza).
- Boty - jeżdżą po gridzie i oczywiście mogą wyjąć jeden kontener z modułu, na którym stoją oraz go przewozić. Mogą go też zostawić w stacji odbiorczej.

W gridzie mamy 3 typy konstrukcji modułów, które różnią się od siebie prędkością, z którą może po nich jeździć bot oraz czasem podnoszenia kontenera z n-tej warstwy. Moduły różnych typów są ze sobą kompatybilne, a więc możemy budować heterogeniczne gridy. Dodatkowo O oznacza, że dany moduł jest aktualnie wyłączony, a boty nie mogą na niego wjeżdżać.

Typ	Opis	Czas przejazdu	Czas wyjmowania
H	**H**igh speed transit - optimized	0,5 [s]	3n + 4 [s]
B	**B**alanced	1 [s]	2n + 2 [s]
S	**S**torage access time - optimized	2 [s]	n + 1 [s]
O	**O**ut of service	-	-

Czas, w jakim bot przejeżdża pomiędzy dwoma modułami, to wolniejszy z “czasów przejazdu” modułów w danej parze, Np. Jeśli bot musi przejechać między modułem B a B, zajmie to 1s. Jeśli musi przejechać między S a H, zajmie to 2s.

Zadanie do rozwiązania

Znając konfigurację magazynu, znajdź najszybszą (czasowo) trasę dla bota, aby z wybranego miejsca startowego na gridzie (x_s, y_s) załadował wybrany produkt p i dostarczył go do stacji odbiorczej na współrzędnych (x_f, y_f).

Wejście

Aby uruchomić program, należy podać dwa argumenty, będące nazwami plików wejściowych odpowiednio z konfiguracją magazynu oraz drugi z zadaniem dla bota.

Plik z konfiguracją magazynu

1. linia: `X:int Y:int N:int` - trzy liczby określające rozmiar gridu oraz warstwy.
następnie: Y linii o długości X, składających się z znaków: {H, B, S, O} i definiujące grid.
następnie: Linie określające współrzędne kontenerów z danym produktem w formacie: `produkt:String współrzędna-x:int współrzędna-y:int warstwa-n:int`, np.

```
P1 2 4 0
P2 0 5 3
```

Pełny przykład gridu 4x3, gdzie przechowujemy 3 kontenery z dwoma różnymi produktami: grid-1.txt

```
4 3 3
HHSH
HBHH
HHOS
P1 3 2 1
P1 0 2 2
P2 1 1 0
```

Zwróć uwagę, że stacja odbiorcza nie jest tutaj zdefiniowana, z punktu widzenia konstrukcji gridu, nie różni się ona niczym od modułu magazynowego.

Plik z zadaniem dla bota

linia 1: `x_s:Int y_s:Int` - określające miejsce startowe bota, np. `2 1`
linia 2: `x_f:Int y_f:Int` - określające współrzędne stacji odbiorczej, np. `0 10`
linia 3: `product:String` - definiujący produkt do dostarczenia, np. `P1`

Przykładowy plik job-1.txt
1 1 0 0 P1

Wyjście

Długość trasy (liczba przejazdów bota między modułami)
Całkowity czas dostarczenia produktu do stacji odbiorczej w formacie `"%.1f"`
Kolejne współrzędne modułów trasy, po jednej parze w jednej linii (razem z początkowym)

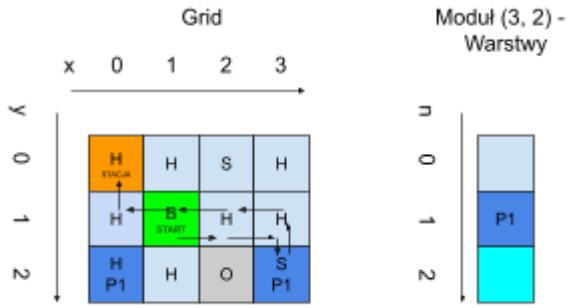
Przykład

Uruchamiając program dla przykładowych danych z wejścia:
`java -jar botrouteplanner.jar grid-1.txt job-1.txt`

Spodziewamy się następującego wyniku na standardowym wyjściu:

```
8
10.5
1 1
2 1
3 1
3 2
3 1
2 1
1 1
0 1
0 0
```

Ilustracja do rozwiązania



- Podpowiedź: na 10,5s całkowitego czasu składa się 3,5s dojazdu z (1,1) do (3,2), 2s załadunku oraz 5s dojazdu z (3,2) do (0,0).
- Uwaga: Czasami jest wiele możliwości rozwiązania, tutaj przedostatnim modułem trasy może być też (1, 0)
- W paczce z zadaniem znajduje się również drugi, większy przykład wraz z wynikiem

Dodatkowe zadanie otwarte

Dodaliśmy w naszym gridzie możliwość zamontowania wielu stacji odbiorczych. Nasz bot może dowieźć produkty do dowolnej stacji odbiorczej. Jak zmodyfikować nasz algorytm, tak aby bot wybrał stację, do której dojedzie najszybciej? Jak wpłynie to na złożoność obliczeniową? (tylko analiza pisemna, maksymalnie 2000 znaków)

Nasze oczekiwania

- W odpowiedzi spodziewamy się archiwum .zip lub .tar.gz z: kodem źródłowym aplikacji, możliwym do uruchomienia jarem oraz plikiem z odpowiedzią na pytanie.
- Można korzystać z dowolnych bibliotek, należy jednak wiedzieć co robią.
- To na co zwracamy uwagę: poprawność wyników, czytelność kodu, jakość projektu (chcielibyśmy aby kod, a przynajmniej kluczowe fragmenty, były pokryty testami). Poprawność, zwięzłość i zrozumiałość odpowiedzi na zadanie otwarte.
- Możesz uzyskać dodatkowe punkty za użycie narzędzia do budowania projektu oraz stworzenie pliku README wraz z kluczowymi informacjami.