AUFGABENSTELLUNG

Allgemeine Informationen

Die Aufgabenstellung ist als *Einzelarbeit* innerhalb von 2,5 Stunden zu lösen. Wie im realen Programmieralltag ist es erlaubt, das Internet zur Recherche zu verwenden. Bei Fragen zur Problemstellung, wende dich bitte an einen der KNAPP-Betreuer.

Einleitung

KNAPP zählt zu den führenden Technologieunternehmen für Automatisierungslösungen und Software für Distribution und Produktion. Kunden auf der ganzen Welt aus unterschiedlichen Branchen wie zum Beispiel Gesundheitswesen, Onlinehandel, Lebensmittelhandel, Mode- und Textil-Branche bis hin zum Automotive-Bereich vertrauen auf die intelligenten Lösungen von KNAPP. Jede dieser Branchen hat dabei ihre Besonderheiten, denen KNAPP durch maßgeschneiderte Systemlösungen gerecht wird.

Aufgabe

Viele unserer Kunden nutzen das Manual Picking System (MPS) um Produkte mit geringer Nachfrage effizient im Lager bedienen zu können. Die Kommissionierung erfolgt in einem manuellen und einem automatischen Schritt, es wird daher auch halbautomatisch genannt.

Deine Aufgabe ist es, den Algorithmus für Abarbeitung der Kundenlieferungen für den manuellen Schritt zu implementieren.

Die Aufgabe ist eine Optimierungslösung, versuche zuerst die Aufgabe zu lösen und dein Ergebnis danach Schritt für Schritt zu verbessern.

Das Manuelle Picking System besteht aus

- Regalen (*shelves*), in denen Produkte gelagert werden.
- Schalen (*bin*), in denen Produkte für Kundenbestellungen (*order*) abgelegt werden.
- Sammelband und Übergabestelle zum automatischen Sammeln aller Produkte eines Kundenauftrages und Übergeben in einen Versandkarton. Diese Komponenten werden in der Lösung nicht benötigt und sind nicht abgebildet.

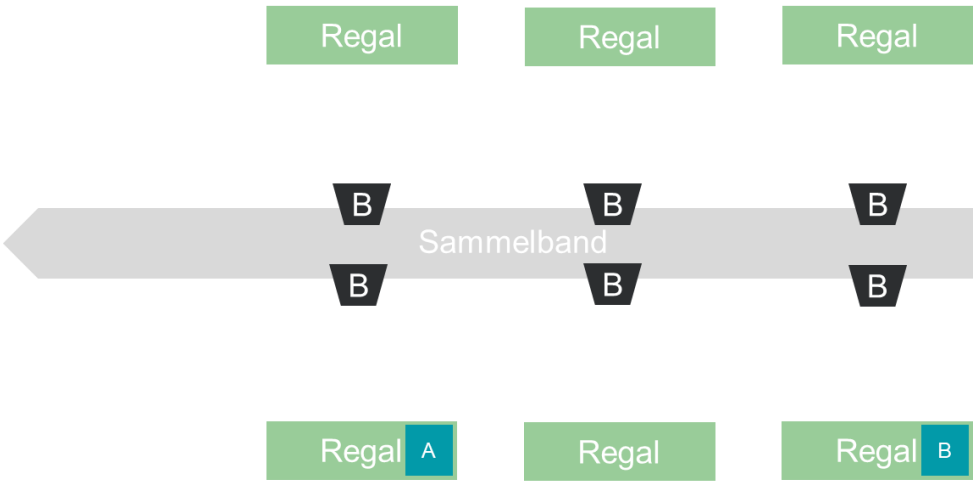

Abbildung 1 - Aufbau MPS

Ein Kunde bestellt ein oder mehrere Produkte in einer Kundenbestellung. Diese Produkte werden zuerst aus den Regalen in Schalen gelegt. Wenn alle Produkte einer Kundenbestellung in Schalen sind, wird die Bearbeitung der Kundenbestellung für den manuellen Bereich beendet. Der MPS wirft alle Produkte aus den Schalen auf das Sammelband. Dieses transportiert die Produkte zur Übergabestelle, wo sie in einen Versandkarton übergeben.

Anfangszustand

- Alle zu bearbeitenden Kundenbestellungen erhältst du mit der Methode (*input.getAllOrders()*).
- Alle Schalen sind leer.
- Die aktuelle Position ist rechts auf Länge 0.

Die Abarbeitung einer Kundenbestellung erfolgt in folgenden Schritten:

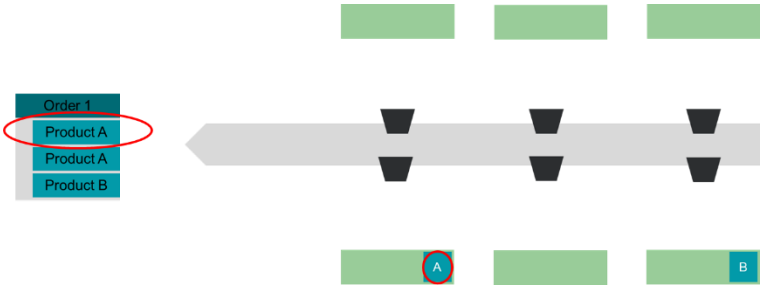

Abbildung 2 - Auswählen von Kundenbestellung und Produkt

Wähle zuerst die Kundenbestellung aus die Du bearbeiten willst. Alle Kundenbestellungen erhältst Du mit der Methode `warehouse.getAllOrders()`.

Aus dieser Kundenbestellung wählst du ein noch benötigtes Produkt aus (`order.getOpenProducts()`). Für dieses Produkt erhältst Du die Regale mit `warehouse.getProductShelves()` und der Suche nach dem Produkt in der Rückgabe.

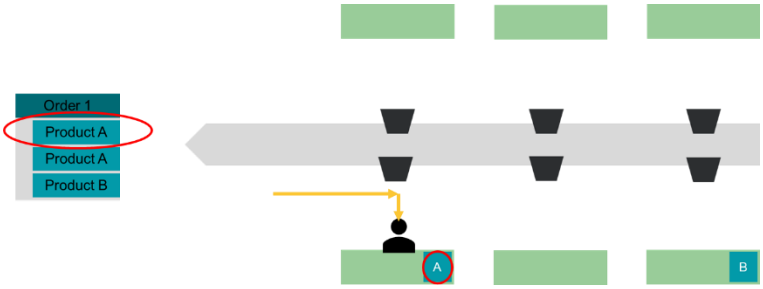

Abbildung 3 - Aufnehmen des Produktes

Nun wird das Produkt aus diesem Regal aufgenommen (`pickProduct()`). Dabei werden unendlich viele Produkte aufgenommen.

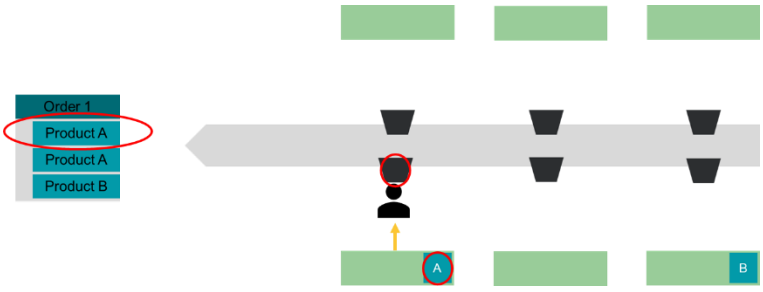

Abbildung 4 - Schale zuweisen und Produkte ablegen

Wähle eine Schale für die Kundenbestellung aus. Alle Schalen erhälts Du mit `warehouse.getAllBins()`, suche aus diesen eine noch nicht zugeordnete, passende Schale aus und ordne dieser die Kundenbestellung mit `warehouse.assignOrder()` zu.

Lege das Produkt in eine zugeordnete Schale mit `putProduct()` ab. Dabei wird immer ein einzelnes Produkt in die Schale gelegt.

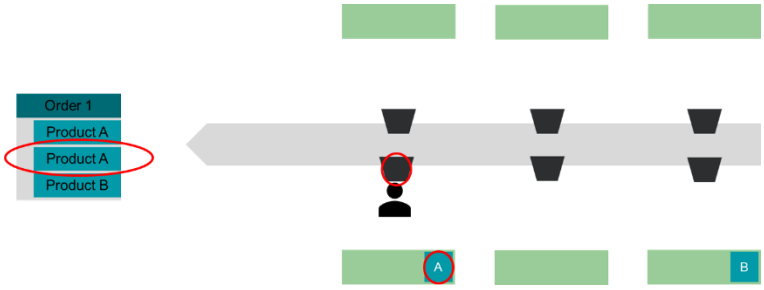

Abbildung 5 - Bearbeiten nächstes Produkt

Wenn dasselbe Produkt in der Kundenbestellung ein weiteres Mal benötigt wird, so lege es nochmals in eine Schale ab.

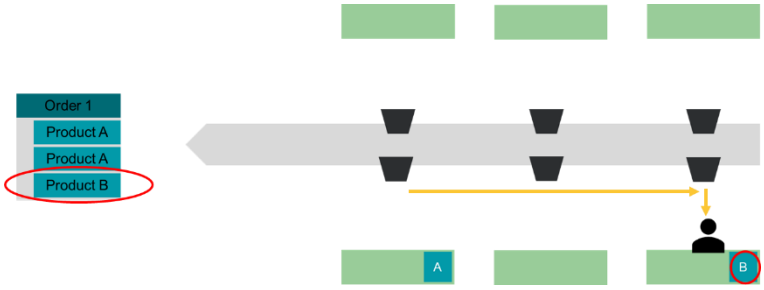

Abbildung 6 - Bearbeiten weiteres Produkt 1

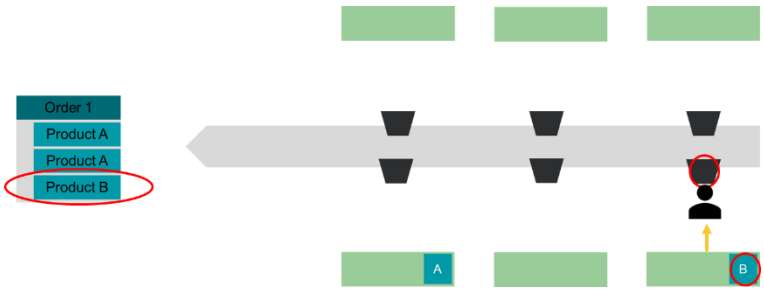

Abbildung 7 - Bearbeiten weiteres Produkt 2

Sind noch weitere Produkte offen, so wiederhole die Schritte wie oben

- Auswahl des Produktes
- Auswahl des Regals
- Aufnehmen des Produktes
- Auswahl der Schale
- Ablegen des Produktes

bis alle Produkte der Kundenbestellung erfüllt sind.

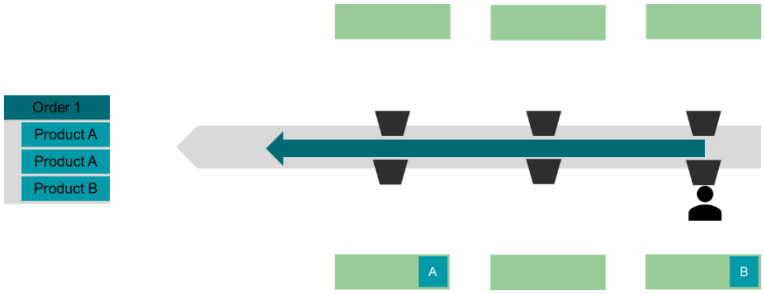

Abbildung 8 - Beenden einer Kundenbestellung

Danach beendest Du die manuelle Bearbeitung der Kundenbestellung mit `finishOrder()`. Dadurch werden – durch den KNAPP Code im Hintergrund – alle Schalen dieser Kundenbestellung auf das Sammelband abgegeben und über das Sammelband an der Übergabestelle in einen Versandkarton gefüllt. Nach dem Beenden der Kundenbestellung sind alle zugeordneten Schalen wieder frei und können einer anderen Kundenbestellung zugeordnet werden.

KOSTEN

Kosten entstehen durch die Bewegung, wenn Tätigkeiten (*pickProduct*, *putProduct*) an unterschiedlichen Positionen ausgeführt werden.

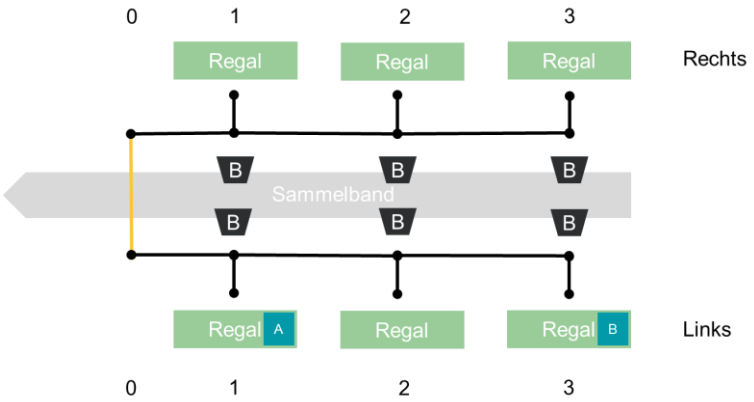

Abbildung 9 - Kosten für Bewegungen

Die Kosten entstehen durch

- Bewegung entlang der Längsseite des Sammelbandes.
- Wechsel zwischen Regal und Schale.
- Wechsel der Seite
    - Der Wechsel der Seite erfolgt immer an der Übergabestelle.
    - Die Kosten für eine Bewegung mit Seitenwechsel ist die Summe aus
        - Der Bewegung von der aktuellen Position zur Übergabestelle
        - Dem Seitenwechsel
        - Der Bewegung von der Übergabestelle zur geforderten Position

ALLGEMEINES

- Alle Kundenbestellungen stehen von Anfang an zur Verfügung.
- Eine begonnene Kundenbestellung kann nur durch Beenden der Kundenbestellung aus den Schalen entfernt werden.
- Eine Kundenbestellung kann auch ohne aufgenommenes Produkt einer Schale zugewiesen werden,
- Beim Aufnehmen der Produkte (*pickProduct*) werden unendlich viele Produkte aufgenommen. Davor aufgenommene Produkte werden durch den KNAPP

Code zurückgeräumt. Produkte können mehrmals in dieselbe oder unterschiedliche Schalen abgegeben (`putProduct`) werden.

- Es gibt unendlich viele Stück eines Produktes in den Regalen.
- Eine Schale ist groß genug, um alle Produkte einer Kundenbestellung aufzunehmen.
- Es können alle Kundenbestellungen vollständig bearbeitet werden.
- Einige Produkte sind in mehreren Regalen vorhanden.

WICHTIGE METHODEN

```
input.getAllOrders()
```

Liefert die Liste der Kundenbestellungen (Order).

```
order.getAllProducts()
order.getOpenProducts()
```

Liefert alle oder die noch zu bearbeitenden Produkte einer Kundenbestellung.

```
warehouse.getAllBins()
```

Liefert alle Schalen, die es im System gibt.

```
workStation.getProductShelves()
```

Liefert alle Regale, die es im System gibt als Map. Das Produkt, dass Du benötigst, musst Du aus dem Rückgabewert auswählen.

```
workStation.pickProduct(Shelf shelf, String product)
```

Nimmt ein Produkt aus einem Regal auf.

- *InvalidShelfException*
  Das angegebene Regal existiert nicht.
- *InvalidProductException*
  Das angegebene Produkt existiert nicht.
- *ProductNotFoundAtShelfException*
  Das geforderte Produkt existiert in diesem Regal nicht.

`warehouse.assignOrder(Order order, Bin bin)`

Weist eine Schale einer Kundenbestellung zu.

- *InvalidOrderException*
  Die angegebene Kundenbestellung existiert nicht.
- *InvalidBinException*
  Die angegebene Schale existiert nicht.
- *OrderAlreadyReleasedException*
  Die angegebene Kundenbestellung wurde bereits beendet.
- *MpsBinAlreadyAssignedException*
  Die angegebene Schale ist bereits einer anderen Kundenbestellung zugewiesen.

`warehouse.putProduct(Bin bin)`

Gibt ein Stück des zuletzt aufgenommenen Produktes in die angegebene Schale ab.

- *InvalidBinException*
  Die angegebene Schale existiert nicht.
- *NoSuchProductInOrderLeftException*
  Das aktuelle Produkt wird in der, der Schale zugeordneten Kundenbestellung nicht mehr benötigt.
- *NoSuchProductInOrderException*
  Das aktuelle Produkt existiert in der Kundenbestellung, die der Schale zugeordnet ist, nicht.
- *NoOrderAssignedToMpsBinException*
  Der angegebenen Schale ist keine Kundenbestellung zugeordnet.

`warehouse.finishOrder(Order order)`

Beendet die manuelle Bearbeitung der Kundenbestellung. Dadurch werden alle Schalen, die der Kundenbestellung zugeordnet sind, freigegeben.

- *InvalidOrderException*
  Die angegebene Kundenbestellung existiert nicht.
- *OrderNotCompletelyPickedException*
  Die Kundenbestellung enthält noch offene Produkte.

`warehouse.getCurrentPosition()`

Liefert die aktuelle Position (letztes `pickProduct` oder `putProduct`).

`warehouse.calcCost(Position pos1, Position pos2)`

Liefert die Kosten, die für die Bewegung von pos1 zu pos2 entstehen.

`warehouse.getOrderAssignedToBin(Bin bin)`

Liefert die dem angegeben bin zugewiesene Kundenbestellung. `null` wenn keine zugewiesen ist.

warehouse.calcCost(Position pos1, Position pos2)

Liefert die Kosten, die für die Bewegung von pos1 zu pos2 entstehen.

ERLÄUTERUNGEN

Kundenbestellung (Order)

Eine Kundenbestellung die ausgeliefert werden soll. Sie beinhält eine *unsortierte* Liste von Produkten. Wenn der Kunde mehr als ein Stück bestellt hat, so sind die Produkte mehrfach in der Liste vorhanden.

Alle Kundenbestellungen stehen von Anfang an zur Verfügung, es gibt keine vorgegebene Reihenfolge, in der diese oder die beinhalteten Produkte abzuarbeiten sind.

Produkt (Product)

Ein Produkt ist das Erzeugnis, das der Kunde bestellt hat. Die Produkte werden von Erzeugern an das Lager geliefert, dort eventuell zwischengelagert, für einen Auftrag kommissioniert und danach an den Kunden versendet.

Ein Produkt ist als dessen Produktcode, einem String dargestellt. Es existiert dafür keine eigene Klasse.

Regale (Shelves)

Ein Regal bietet den Lagerplatz für mehrere Produkte. Es sind alle Produkte eines Regales gleich gut zu erreichen. Der Platz für ein Produkt im Regal ist unendlich, es stehen unendlich viele Stück eines Produktes zur Verfügung.

Ein Regal besteht aus

- Einem Code zur Identifikation (*shelf.getCode()*).
- Der Position an der das Regal steht (*shelf.getPosition()*).
- Der Collection an Produkten (*String*s) die in diesem Regal gelagert sind (*shelf.getProducts()*).

Schale (Bin)

Eine Schale buffert Produkte aus einer Kundenbestellung. Eine Schale kann unendlich viele Produkte beinhalten, aber nur einer einzelnen Kundenbestellung zugeordnet sein

Eine Schale besteht aus

- Einem Code zur Identifikation (`bin.getCode()`).
- Der Position, an der die Schale steht (`bin.getPosition()`).

Position (Position)

Eine Position gibt einen Standort innerhalb des Systems an. Eine Schale besteht aus

- Der Seite (`side`)
- Dem Abstand in Längsrichtung von der Übergabestelle aus (`lengthwise`).

MUSS-BEDINGUNGEN

- Nur offene Produkte der aktuell zugewiesenen Kundenbestellung dürfen in eine Schale abgegeben werden.
- Alle Produkte einer Kundenbestellung müssen in einer zugewiesenen Schale liegen, bevor diese Kundenbestellung beendet werden kann.

OPTIMIERUNGEN

Versuche zuerst die Aufgabe so einfach wie möglich zu lösen. Danach kannst du deine Aufgabe weiter Optimieren

- Ein Produkt kann mehrfach abgelegt werden, ohne es erneut aus dem Regal holen zu müssen.
- Eine Kundenbestellung kann mehreren Schalen zugewiesen werden.
- Eine Kundenbestellung muss nicht ohne Unterbrechung abgearbeitet werden.
- Produkte sind teilweise in mehreren Regalen vorhanden.

BEWERTUNGSSCHEMA

- Es werden nur auf den Bewertungsserver hochgeladene Ergebnisse gewertet.
- Die Resultate werden am Server, mit den dort hinterlegten Algorithmen, errechnet.
- Die Punkte errechnen sich aus
    - Den gesamten Wegkosten
    - Einem Penalty, wenn nicht alle Kundenbestellungen abgearbeitet wurden
    - dem Abgabezeitpunkt Deines besten Ergebnisses (früher ist besser)

Die Gewichtung zwischen Abgabezeitpunkt und restlichen Kosten ist dabei so gewählt, dass in der Regel eine bessere Lösung auch bei späterer Abgabe ein besseres Ergebnis bedeutet.

Bei mehreren Abgaben (Uploads) zählt immer die beste Abgabe zu dem Zeitpunkt, an dem diese getätigt wurde. Wenn Du nach dem Upload einer Lösung weiterarbeitest und durch Deine Änderungen das Ergebnis schlechter wird, bleiben spätere Abgaben unberücksichtigt.

ABGABEMODUS

Zur Beurteilung des Ergebnisses muss die vom KNAPP-Code automatisch erzeugte Datei `upload-xxxxx.zip` über die Abgabeseite hochgeladen werden. In diese Datei wird dein Source automatisch mitverpackt. Dieser Source muss das Ergebnis erzeugt haben. KNAPP behält sich hier Kontrollen vor.

Du kannst alle Code Teile modifizieren, die Ergebnisdatei (`result.csv`) wird von uns interpretiert und darf daher im Format nicht verändert werden.

UPLOAD WEBSITE

Unmittelbar nach dem Upload erhältst du ein detailliertes Feedback zu deiner Abgabe.

Im Feedback siehst du die Probleme, die noch in deiner Abgabe vorhanden sind, und deinen Score für diesen Upload. Je *niedriger* dieser Wert ist, desto besser ist das Ergebnis.

TIPPS

- Versuche mit deiner Software das Ergebnis zuerst mit einfachen Strategien zu verbessern und arbeite danach an weiteren Optimierungen.
- Schau dir den von uns vorgegeben Code an und prüfe, ob du Teile wiederverwenden oder erweitern kannst.
- Du kannst alle Teile des Source-Codes verändern. Die Abgabedatei muss dem vorgegebenen Format entsprechen.
- Lade öfters hoch - es wird nur die beste Abgabe bewertet.

CODE-DETAILS

- Name und Institution setzen.
- Einsprungspunkt: `Solution::run()`
- Die Klassen im Package `core` sind KNAPP Hilfsklassen und sind für die Lösung nicht von Bedeutung.