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

Eines der automatischen Kommissioniersysteme das KNAPP für Läger aber auch für den Einsatz direkt in Filialen anbietet ist der KNAPP-Store. Dieser ist optimiert für die Kommissionierung und Lagerung von geringen Stückzahlen von vielen verschiedenen Produkten und bietet den Vorteil, eine große Anzahl an Produkten auf geringem Platz lagern zu können.

Der KNAPP-Store besteht aus einem speziellen Lagerort für die Einlagerung von Produkten in das System, Lagerorten zur Lagerung der Produkte, einem speziellen Lagerort zur Ausgabe der Artikel und einem Roboter der Produkte zwischen den einzelnen Lagerorten bewegt.

Deine Aufgabe ist es nun, den Algorithmus für Abarbeitung der täglichen Arbeit, des Einlagerns und der Auftragsabarbeitung zu implementieren. Ziel ist, die Aufträge mit so wenig Kosten wie möglich zu bearbeiten.

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

Ablauf

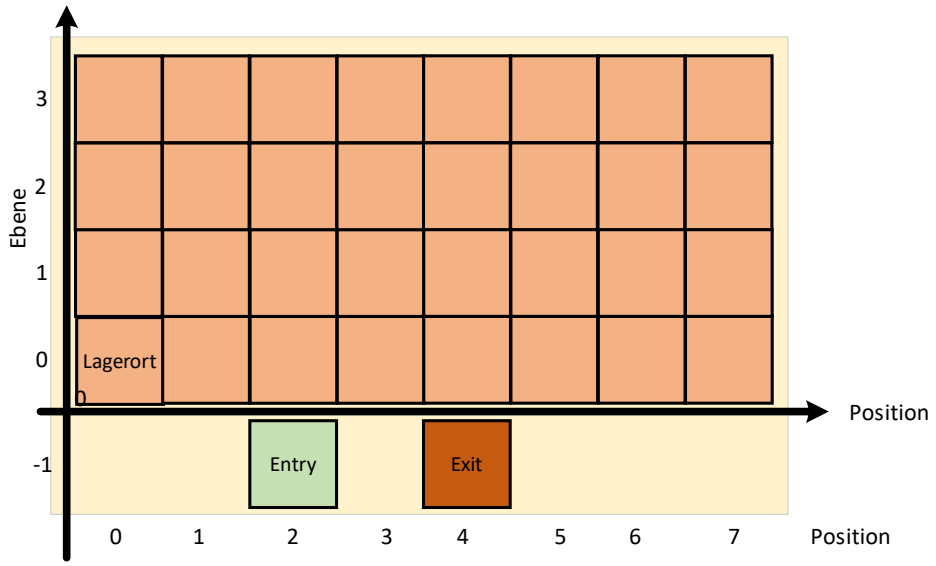

Abbildung 1 – Aufbau des KNAPP-Store (Schema)

Vor der Eingangsposition liegt eine Reihe von Produkten, die der Reihe nach abgearbeitet werden müssen (*`warehouse.GetProductsAtEntry()`*). Zusätzlich gibt es eine Reihe von Aufträgen (*`warehouse.GetRemainingOrders()`*) die diese Produkte benötigen und abgearbeitet werden sollen. Die Handhabung der Produkte erfolgt über einen Roboter.

Um ein Produkt aufzunehmen, fährt der Roboter zur Eingangsposition (*`entryLocation`*) und nimmt dann ein Produkt auf (*`robot.PullFrom(entryLocation)`*).

Dieses kann dann auf einem anderen Lagerort abgelegt werden (*`robot.PushTo(xxx)`*). Der Roboter kann es mit (*`robot.PullFrom(xxx)`*) wieder aufnehmen und für einen Auftrag an der Ausgabeposition abgeben (*`robot.PushTo(exitLocation)`*) werden.

Die Reihenfolge der Produkte an der Eingabeposition ist vorgegeben, die Produkte müssen in dieser Reihenfolge geholt werden. Wenn diese nicht sofort für den aktuellen Auftrag gebraucht werden, so müssen diese auf einem Lagerort im KNAPP-Store gelagert werden, bis es benötigt wird.

Die Auswahl welches Produkt auf welchen Lagerort gelegt oder von welchem Lagerort genommen wird trifft dein Code. Diese Auswahl hilft dir bei der weiteren Optimierung deiner Lösung.

Lagerung von Produkten

Die Handhabung der Produkte durch den Roboter erfolgt bei der Abgabe durch einen Schieber, durch den die Produkte abgeschoben werden.

Bei der Aufnahme durch den Roboter fährt dieser seitliche Greifer aus mit denen das Produkt geklemmt wird und das Produkt auf den Roboter gezogen wird.

Durch diese Mechanik können *sowohl* am Roboter als auch am Lagerort mehrere Produkte gelagert werden.

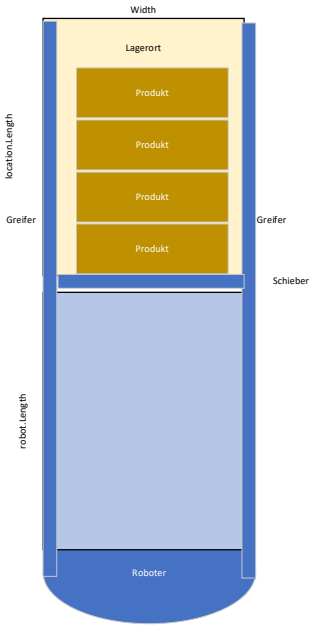

Abbildung 2 - Lagerung - Aufbau Roboter

Die Anzahl der Produkte ist begrenzt durch die Summe der Länge der Produkte. Diese muss kleiner sein als die Länge des Lagerortes. Die Länge des Roboters und des Lagerortes ist gleich.

Pro Aufruf von `PullFrom` oder `PushTo` wird ein Produkt bewegt, sollen mehrere aufgenommen oder abgeben werden, so muss die Methode mehrmals hintereinander aufgerufen werden.

Bedingt durch den mechanischen Aufbau muss die Breite (Width) der Produkte auf dem Roboter gleichbleiben oder größer werden da diese sonst nicht mehr bewegt

werden können. Dies gilt bei der Aufnahme von der Eingangsposition und den Lagerorten.

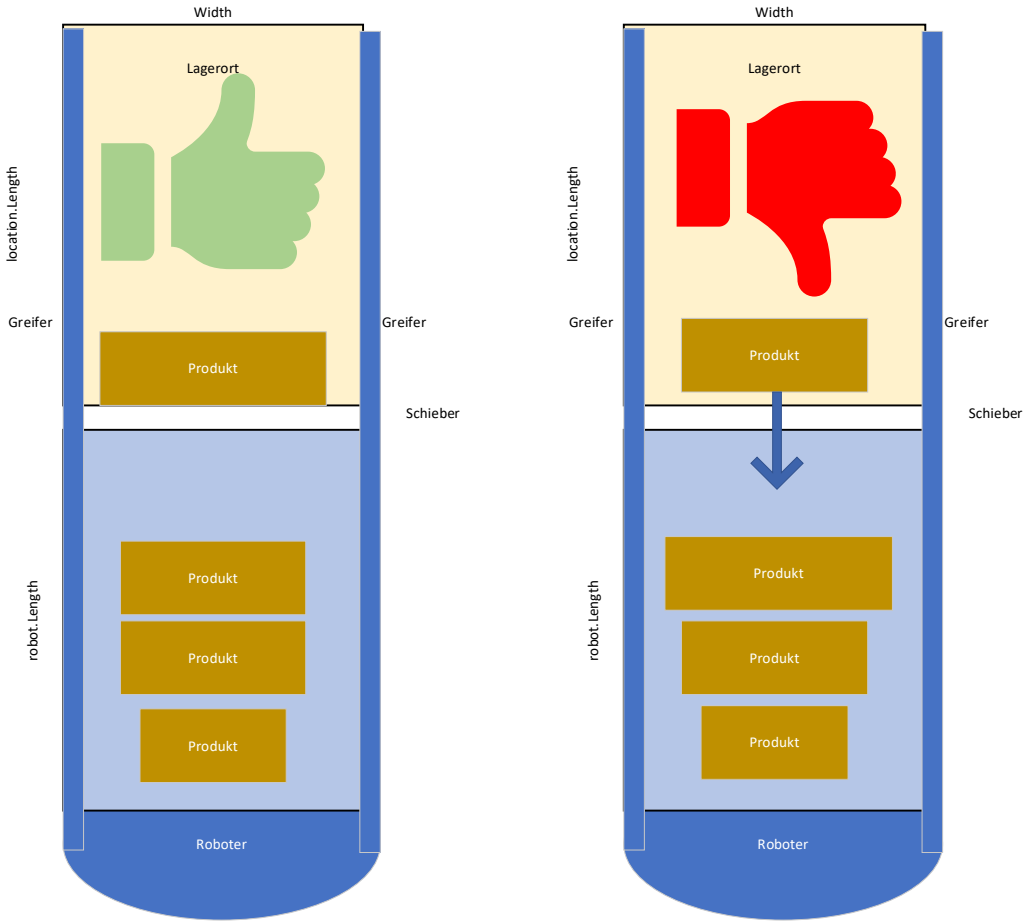

Abbildung 3 - Lagerung - PullFrom OK und nicht OK

WICHTIGE METHODEN

warehouse.GetRemainingProductsAtEntry()

Gibt eine sortierte Liste der Produkte zurück, die noch am Einlagerungspunkt warten. Das erste Produkt in dieser Liste wird durch das nächste `PullFrom(entryLocation)` auf den Roboter geladen.

Warehouse.HasNextOrder()

Gibt es noch einen Auftrag der nicht bearbeitet wurde?

warehouse.NextOrder()

Prüft, ob der aktuelle Auftrag abgeschlossen ist und schaltet den nächsten Auftrag aktiv.

order.GetProducts()

Liefert eine Liste der im Auftrag bestellten Produkte zurück.

storage.GetLocation(level, position)

Gibt den Lagerort an der geforderten Position zurück.

storage.GetAllLocations()

Gibt eine Liste mit allen Lagerorten (außer der Eingabe- und Ausgabeposition) im KNAPP-Store zurück.

robot.PullFrom(location)

Nimmt ein Produkt vom angegebenen Lagerort auf den Roboter auf.

robot.PushTo(location)

Gibt ein Produkt vom Roboter auf den angegebenen Lagerort ab.

ERLÄUTERUNGEN

Auftrag (Order)

Ein Auftrag umfasst alle Produkte, die an einen Kunden versendet werden. Der Auftrag kann 1 – n Produkte enthalten, die einzelnen Produkte können ein oder mehrmals im Auftrag vorkommen.

Wenn die Bearbeitung eines Auftrages begonnen wird, so muss diese abgeschlossen werden.

Alle Aufträge stehen von Anfang an zur Verfügung, die Reihenfolge, in der die Aufträge bearbeitet werden müssen, ist vorgegeben.

Produkte (Product)

Ein Produkt ist das Erzeugnis, dass der Kunde in einem Auftrag 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 hat

- Einen Code der es eindeutig identifiziert
- Eine Länge (Length)
- Und eine Breite (Width)

Dabei gilt: Länge ist die Abmessung in der Richtung, in der die Produkte in einen Lagerort gelegt werden. Die Abmessung der Länge des Produktes und Länge des Lagerortes sind in gleicher Richtung gemessen.

Lagerort (Location)

Ein Lagerort ist ein Platz innerhalb des KNAPP-Store an dem die Produkte gelagert werden können. Ein Lagerort befindet sich an einer bestimmten Position im Regal (Level, Position) und kann eine gewisse Menge an Produkten aufnehmen die

Ein Lagerort hat:

- Typ (Entry, Exit, Storage)
- Ebene (Level)
- Position
- Länge (Length)

Roboter (Robot)

Der Roboter bewegt die Produkte innerhalb eines KNAPP-Stores. Er kann ein Produkt von einem Lagerort aufnehmen und ein aufgenommenes Produkt auf einen Lagerort abgeben. Wenn mehrere Produkte von einem Lagerort aufgenommen oder auf diesen abgegeben werden sollen, so muss dies hintereinander erfolgen.

Der Roboter hat, wie ein Lagerort, eine Länge[1], die die Anzahl an Produkten bestimmt, die er zur selben Zeit mit sich führen kann.

[1] Die Länge des Roboters und des Lagerortes ist gleich

ALLGEMEINES

- Alle Daten (Aufträge, Produkte) stehen von Anfang an zur Verfügung.
- Die Reihenfolge, in der die Produkte angeliefert werden, ist vorgegeben.
- Die Reihenfolge, in der die Aufträge abgearbeitet werden, ist vorgegeben.
- Die Produkte eines Auftrages können an der Ausgabeposition in beliebiger Reihenfolge abgeben werden.
- Wenn mehrere Operationen hintereinander auf denselben Lagerort ausgeführt werden, entstehen keine Kosten durch die Fahrt.
- Es gibt mehr Lagerorte im KNAPP-Store als Produkte, die auf Einlagerung warten.
- Es müssen nicht alle Produkte eingelagert werden.
- *foreach* und *iteratoren* funktionieren nicht für Collections die im Hintergrund verwendet werden (ConcurentModification). Dies sind zB RemainingXXX. Hier muss die Schleife mit einer Prüfung auf leer selbst erstellt werden.
- Die Ergebnisdatei wird automatisch erstellt.

MUSS-BEDINGUNGEN

- Bei der Aufnahme oder Abgabe von Produkten muss immer ein Produkt vorhanden sein.
- Die insgesamte Länge der auf einem Lagerort oder dem Roboter gelagerten Produkte darf nicht größer sein als dessen Länge. Ausgenommen ist hier die Ausgabeposition.
- Der Roboter kann nur aufnehmen, wenn die Breite des aufzunehmenden Produktes größer oder gleich der aktuellen maximalen Breite am Roboter ist.
- Ein begonnener Auftrag muss komplett erfüllt werden, bevor der nächste begonnen werden kann.
- Es kann nur ein für den Auftrag benötigtes Produkt abgegeben werden.
- Die Einlagerung muss in der vorgegebenen Reihenfolge erfolgen.
- Die Abarbeitung der Aufträge muss in der vorgegebenen Reihenfolge erfolgen.

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
    - der gesamt gefahrenen Strecke des Roboters
    - Summe der Distanz zwischen den Lagerorten von aufeinanderfolgenden Operationen (PullFrom oder PushTo)
    - Bestellte Produkte, die sich noch am Einlagerungspunkt befinden
    - Bestellte Produkte, die ein- aber noch nicht ausgelagert wurden
    - einem Penalty, wenn nicht alle Aufträge abgearbeitet wurden
    - dem Abgabezeitpunkt (früher ist besser)

Die Gewichtung zwischen Abgabezeitpunkt und 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.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.

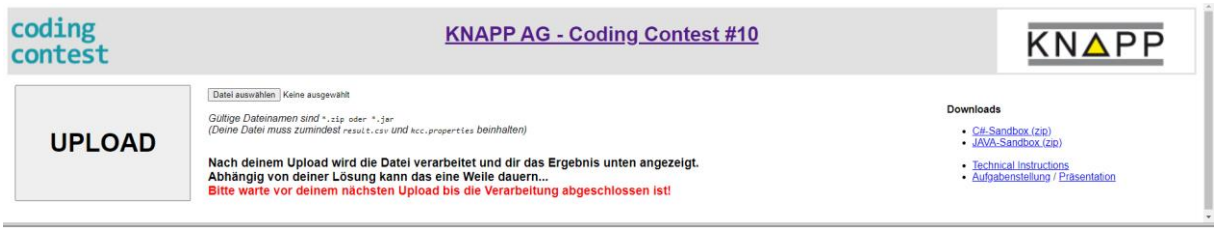

Abbildung 4: Upload-Server (Beispiel)

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.

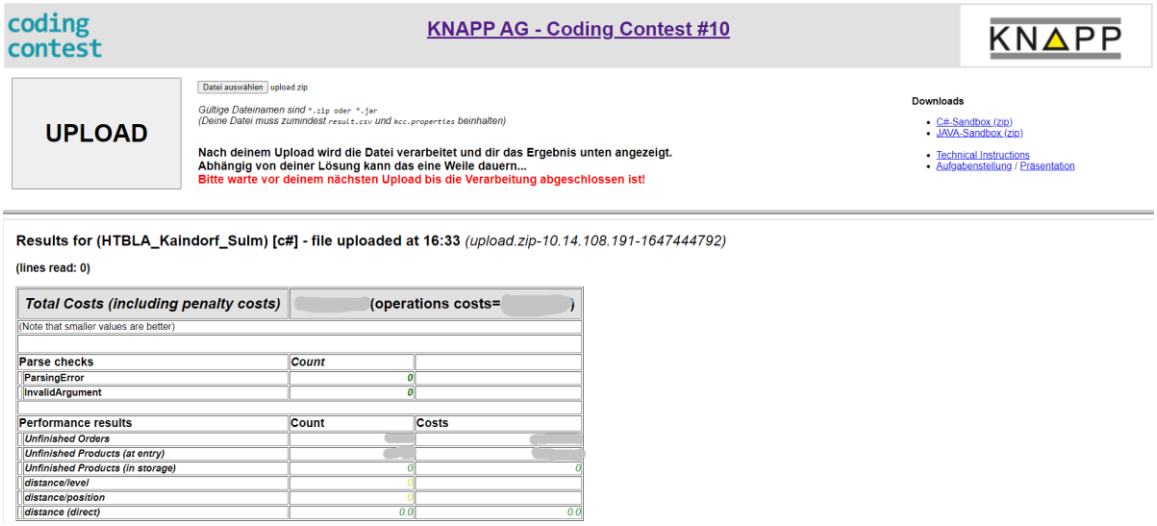

Abbildung 5 Abgabe - Details (Beispiel)

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 jedoch 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

Klassendiagramm

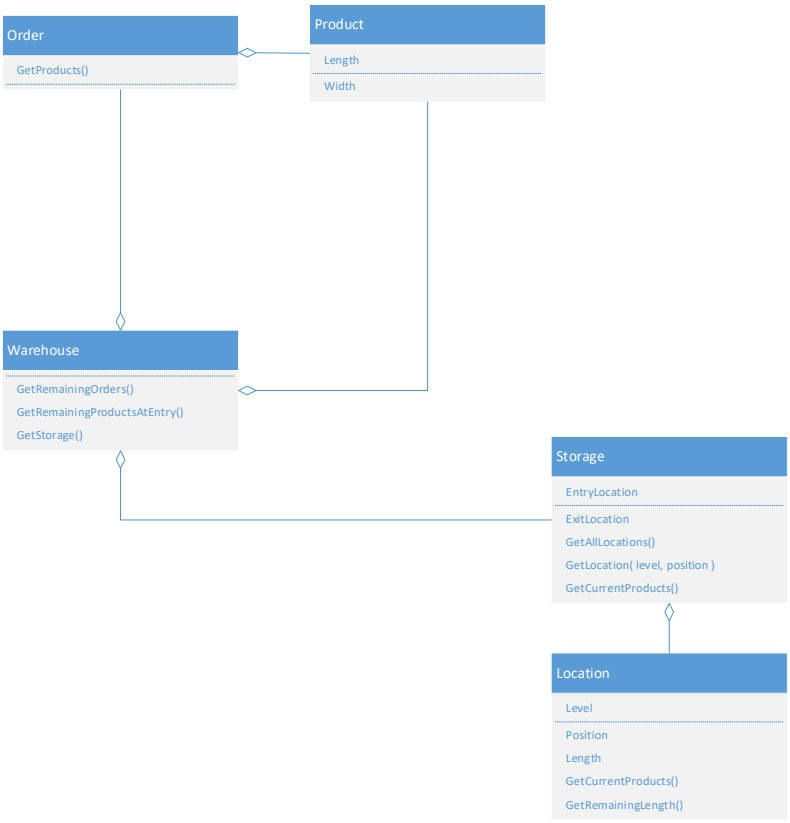

Abbildung 6 - Klassendiagramm