

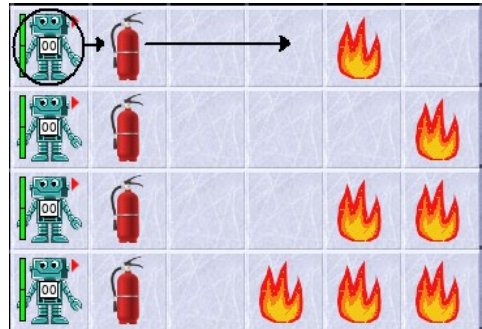


In einem anderen Kraftwerk ist ein Feuer ausgebrochen. Deine ersten Erfolge haben sich rumgesprochen. Daher wird deine RoboRescue-Firma angesprochen, ob sie auch in diesem Fall helfen kann. Natürlich bist du sofort bereit... aber wie löscht man ein Feuer?

## Die Roboter machen vieles immer wieder ...

**ZIEL:** Wiederholungen in Handlungen erkennen und in der Programmiersprache mit dem Schlüsselwort *while* umsetzen können. Methoden mit Parametern benutzen können.

### Aufgaben:



1. **Löschen von Hand:** Der Roboter **AB3** ganz oben links soll den vor ihm stehenden Feuerlöscher aufnehmen, damit auf das Feld vor dem Feuer (nicht auf das Feuer!!!) gehen und es löschen.

Führe dies zunächst von Hand durch. Mit Rechtsklick → bei „**geerbt/inherited von Roboter**“ findest du alle Methoden; auch die Methode **benutze(String name)**. Hier muss man einen String (=Text mit Anführungszeichen) angeben, welchen Gegenstand man benutzen möchte.

2. **Löschen im Code:** Vervollständige dann die Methode **loesche()** im Quelltext, die das Löschen des ersten Feuers für den oberen Roboter anweisen soll.
3. **Unterschiedliche Entfernung:** Die weiteren Roboter darunter sollen ihre **ersten** Feuer in der Reihe ebenfalls mit dem Befehl **loesche()** löschen können. Diese Feuer sind aber in einer anderen Entfernung platziert. Überlege dir, welche Zeilen deiner Methode **loesche()** überarbeitet werden müssen. Worauf muss der Roboter reagieren können?
4. **VorBisFeuer():** Gib jedem Roboter **händisch** den Befehl **vorBisFeuer()** und beobachte, was sie tun.  
Lies danach im Quelltext bei der Methode **vorBisFeuer()** die Anweisungen und versuche sie zu verstehen. **Hinweis:** das Ausrufezeichen bedeutet „nicht“. Setze diese Methode sinnvoll in der **loesche()**-Methode ein, sodass jeder Roboter sein erstes Feuer löschen kann. Teste nun die neue **loesche()**-Methode an **allen vier** Löschrobotern.

5. **Mehrere Feuer löschen:** Die Feuer des 3. und 4. Roboters bestehen aus mehreren Flammen hintereinander (ohne Lücke dazwischen). Schreibe eine **neue Methode loescheReihe()**, die wieder den Feuerlöscher aufnimmt und bis zum Feuer läuft und dann zusätzlich alle Feuer löscht!

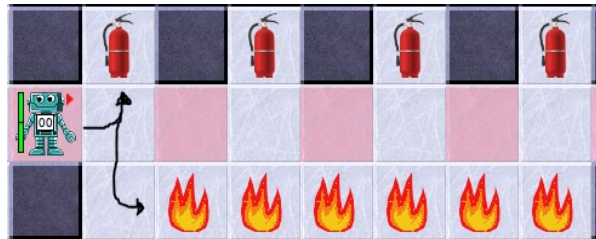


**Tip:** Um alle zu löschen, muss man immer wieder löschen, dann einen Schritt gehen, dann wieder löschen usw.. Das wiederholt man solange, wie nach dem Schritt noch ein Feuer vor dem Roboter ist.



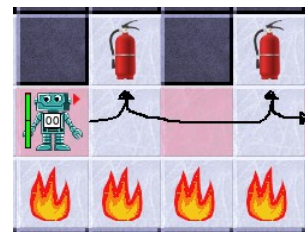
6. **Reichweite testen:** Mit einem Feuerlöscher kommt man nicht weit. Die sind recht schnell leer. Schreibe eine Methode **testeReichweite()**, bei der der Roboter unten links, den ersten Feuerlöscher einsammelt, zur Feuerspur darunter geht und dort so viel Feuer wie möglich löscht.

**Tipp:** Verwende eine while-Schleife, um zu überprüfen, ob der Feuerlöscher noch nicht leer ist. Die Bedingung `(getAnzahl("Feuerloescher")>=1)` testet, ob der Roboter noch mindestens 1 Löscher hat. Dies testet automatisch, ob er leer ist, da er einfach aus dem Inventar des Roboters verschwindet, wenn er verbraucht ist.



7. **Feuerlöscher einsammeln:** Der Roboter unten links soll für ein großes Feuer alle Feuerlöscher einsammeln. Er soll nach vorne bis zur Wand laufen und dabei alle Löscher einsammeln. Implementiere eine Methode **sammleLoescher()**, die zunächst den Roboter bis zur Wand laufen lässt, ohne die Löscher einzusammeln. Die Methode **istVorneFrei()** testet, ob vor dem Roboter keine Wand ist.

**Teste** verschiedene Startpositionen, indem du neue AB3-Roboter erzeugst und auf einem der roten Startfelder platzierst. Hierbei sollen natürlich nur die Feuerlöscher nach dem Startfeld eingesammelt werden.



8. **Homerun:** Lasse den Roboter rechts unten in die Sackgasse („Schneckenhaus“) laufen.  
**a)** Implementiere dazu zunächst die Methode **laufeBisWand()**, die den Roboter geradeaus bis zur nächsten Wand bewegt.

**Tipp:** Verwende eine While-Schleife mit **istVorneFrei()** als Bedingung.

**b)** Implementiere **laufeBisSackgasse()**, indem du zunächst **laufeBisWand()** aufrufst und wiederholend überprüfst, ob man nach links weitergehen kann und anschließend wieder bis zu einer Wand läufst. Du erkennst, ob links keine Wand ist (**!istWandLinks()**) - das **!** steht für „nicht“).

**Bildquellen:** Die verwendeten Bilder des Roboterszenarios sind alle ohne Bildnachweis verwendbar (selbst gezeichnet, Pixabay Lizenz oder Public Domain). Genaue Nachweise: siehe [bildquellen.html](#).

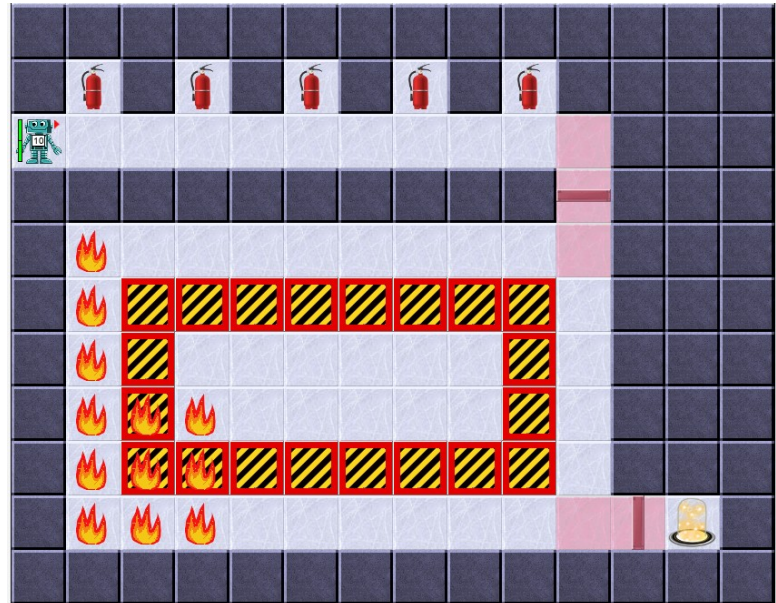


## Einsatz 3: Feuerlöschen im Kernkraftwerk

Der Einsatz 3 kann über Rechtsklick auf die Zeichenfläche aufgerufen werden. Dabei darf **kein Objekt** angeklickt werden.

Implementiere für diesen Einsatz die Methode **insatz3()** im Quelltext. Lass den Roboter zuerst alle Feuerlöscher einsammeln, um dann im Raum unten alle Feuer zu löschen.

**Wichtig:** Nutze dabei die in AB3 zuvor selbst programmierten Methoden und der Roboter verfügt bereits zu Beginn des Einsatzes über 10 Akkus, die benutzt werden müssen, wenn die Energie nicht mehr ausreicht!



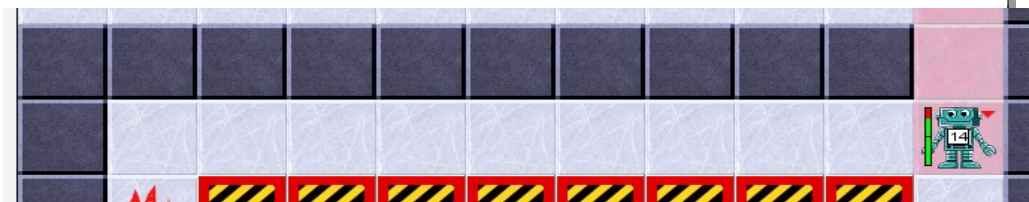
### Tipps, falls es schwierig ist:

- Lösche aus der Methode **loescheReihe()** den Befehl **aufnehmen()**.
- Versuche zunächst, nur die oberste Reihe Flammen zu löschen und danach wieder nach rechts zur Wand zu laufen. Drehe den Roboter wieder so, dass er nach unten blickt (Ausgangsstellung).

Start der 1. Wdh:




Start des 2. Durchlaufs der Wiederholung:



- Sobald man nach dem Zurücklaufen nach rechts auf dem Portal steht, kann die Wiederholung abgebrochen werden. Nutze zur Überprüfung die Methode **istAufGegenstand("Portal")**.

- über den Geschwindigkeitsregler unten rechts kannst du die Ausführungsgeschwindigkeit anpassen!

**Zusammenfassung:** Du kannst Algorithmen formulieren und im Quelltext notieren, die eine oder mehrere Anweisungen solange wiederholen, wie eine Ausführungsbedingung gilt. Als Bedingung eignet sich alles, was wahr oder falsch sein kann. Hierfür eignen sich Methoden wie **istVorneFrei()**, **istVorratLeer()** oder **istWandVorne()** dessen Ja/Nein-Antwort die Wiederholung steuert.

Die Ausführungsbedingung muss im Laufe der Wiederholungen einmal falsch werden, damit es keine Endlosschleife gibt. Falls es dennoch mal eine Endlosschleife gibt, kann man diese mit dem Knopf  unten rechts unterbrechen.