Introduction

Plan :

Contenu

[I – Bilan 1](#_Toc289512702)

[II – Tester un comportement 1](#_Toc289512703)

[1 – Ajouter un nouveau comportement 1](#_Toc289512704)

[2 – Configurer le Roomba 1](#_Toc289512705)

[3 – Tester un comportement 1](#_Toc289512706)

[III – Ajouter des fonctionnalités 1](#_Toc289512707)

[1 – Ajouter une nouvelle forme 2](#_Toc289512708)

[2 – Ajouter une nouvelle interface 2](#_Toc289512709)

[3 – Ajouter un nouveau capteur 2](#_Toc289512710)

# I – Bilan

Afin de respecter le cahier des charges, nous avons modélisé notre programme selon l’UML fourni en annexe.

Les obstacles et les taches peuvent être circulaire, carrés ou rectangulaires. Ces paramètres passent par l’interface graphique ou console.  
L’environnement comprend une arène qui peut être carré ou circulaire. Le paramétrage est fait dans l’interface graphique, mais pour qu’il soit effectif en console, il doit être modifié dans le code.  
Le robot peut être carré, circulaire ou rectangulaire et contenir des capteurs à souhait. Pour simplifier l’exercice, nous avons créé une classe roomba qui crée un roomba aux caractéristiques du vrai roomba. Cependant il est possible de simuler un autre robot en créant une nouvelle classe.  
Le robot contient des capteurs qui sont de trois types : capteur de contact type poussoir, capteur de tache et capteur de distance laser. Il peuvent être ajoutés au robot en modifiant le code, et peuvent servir dans la modélisation de comportement. Il est aussi facile d’ajouter un nouveau type de capteur à partir de la classe abstraite Capteur

Test1

# II – Tester un comportement

## 1 – Ajouter un nouveau comportement

## 2 – Configurer le Roomba

## 3 – Tester un comportement

# III – Ajouter des fonctionnalités

## 1 – Ajouter une nouvelle forme

## 2 – Ajouter une nouvelle interface

## 3 – Ajouter un nouveau capteur