Передовые производственные технологии

2022/23 учебный год

Второй отборочный этап

Представленные задачи являются во многом важными для финала. Во-первых, они знакомят с элементами комплексной задачи, которую им предстоит решить в финале, во-вторых, напрямую тренеруют те навыки, которые им понадобятся для решения задачи финала, в-третьих, учатся комплексно подходить к задаче, применяя при этом современные передовые производственные технологии — моделирование, симуляцию, прототипирование, проектную деятельность, программирование на современных языках.

В рамках финальной задачи им также будет дана задача из области колёсной робототехники, но в другом контексте, а также с несколько другой проблематикой — ориентация колёсной техники в узком замкнутом пространстве и локализация утечки опасных газов.

Командный подход реализован на уровне того, что задача, поставленная перед участниками, является комплексной — без понимания того, какие устройства использовать, где и как устанавливать датчики (задача схемотехника), конструктор не сможет создать корпус для устройства со всеми креплениями для всех устройств, а программист без понимания того, какую информацию и в каком формате будет отдавать ему схемотехник не сможет правильно продумать программу управления.

Ну и, конечно, в рамках комплексной задачи второго этапа, они все вместе реализуют все 3 ступени иерархии системы управления — исполнительный, тактический, стратегический, каждый из которых друг без друга существовать не может.

Больше информации можно найти в отдельной главе — введение, она ниже в тексте задания.

Задачи, также как и на первом этапе, разделены на 3 основных блока — конструирование, схемотехника и программирование, причём компетенции, необходимые для решения задач 2 этапа не изменились — это программирование на языке Python, знание схемотехники, умение программировать Arduino, владение современными средствами автоматизированного проектирования или САПР, комплексный подход к мышлению.

Введение

Поздравляю тебя, участник 2 отборочного этапа олимпиады НТО по профилю «Передовые производственные технологии»! Ты прошел 1 отборочный этап, а, значит, доказал, что можешь выполнять различные несложные задачи. Однако это достаточно скучно, не так ли? Кроме того, все задачки ты решал самостоятельно (самостоятельно, ведь, правда?), а настоящий проект требует слаженной работы целой команды, даже если она небольшая, но очень активная. Итак, давайте теперь проверим ваши навыки по проектированию, но уже в команде!

Проектирование на всех стадиях — разработка, инжиниринг и производство, где производство является лишь конечной стадией — и есть суть передовых производственных технологий. Мы с вами пройдём весь этот путь от начала до конца, но, прежде всего, давайте с вами поработаем над более сложными задачами, которые

позволят погрузиться в мир передовых разработок.

Сегодня практически невозможно представить себе мир без автоматизации в различных сферах человеческой жизни. Мы привыкли к тому, что различные устройства способны очень сильно облегчить нам жизнь, а другие — иногда и спасать их. Это можно делать различными способами — устранением или перемещением объектов, представляющих опасность, и обнаружение людей и различных объектов, и исследование завалов или опасных мест и многое другое. Однако, пожалуй, одной из самых важных задач является поиск и точная локализация источников угрозы. Согласитесь — ведь мало того, что необходимо узнать о том, что опасность присутствует, нужно ещё и предоставить как маршрут к этой точке, так и точное расположение опасности.

Давайте в рамках нашего с вами группового отборочного этапа спроектируем некоторые основные компоненты будущего поисково-спасательного робота. Мы предложем вам решить несколько инженерных задач по конструированию, схемотехнике и алгоритмистике с программированием, которые лягут в основу чего-то большего, чем мы с вами займёмся уже в финале, ну а пока давайте начинать!

На втором этапе в вашу задачу будет входить создание элементов небольшого поисково-спасательного робота. Мы расскажем о его задачах и некоторых ограничениях в работе, однако вне этих ограничений — вы можете поступать так, как считаете нужным и ограничиваться только вашей фантазией.

К нам обратился крупный заказчик, им которого в целях секретности он нас попросил не называть, который, однако, попросил создать небольшого робота, которому предстоит выполнять несложное, но очень важное дело — искать утечки опасных газов. Однако, что сложного, спросите вы в том, чтобы самостоятельно человеку пройти и проверить — есть ли она или нет? И будете правы, задавая такой вопрос, однако поясним:

Во-первых, утечка опасных газов может привести к серьёзным последствиям как для заказчика, так и для человека, который может эту утечку искать, ведь представьте себе, что может произойти, если при утечке, скажем, пропана, которым мы часто пользуемся для того, чтобы разогреть или приготовить еду, возникнет искра? Устройство, а точнее, робота, конечно, будет жалко, но не так, как живых людей.

А во-вторых, место, где планируется искать такую утечку может быть слишком мало даже для небольшого человека, не говоря уже о взрослом.

Итак, вам необходимо совместно продумать и разработать элементы малогабаритного робота, в задачу которого будет входить поиск в замкнутом пространстве места утечки опасного газа, а, по его нахождению, вернуться на исходную позицию.

Конструирование

Задача IV.2.1. (50 баллов)

Темы: конструирование, моделирование, САПР, Fusion360, робототехника.

Условие

Казалось бы — задача конструирования целиком и полностью лежит на конструкторе, однако это заблуждение. И, думаю, вы в этом скоро убедитесь.

Дело в том, что очень много элементов общих для проекта должны сложиться воедино для того, чтобы приступить к конструированию и вот почему: в конструкции робота очень часто приходится предусматривать различные элементы, направленные, например, на то, чтобы в корпусе можно было разместить подобранный схемотехником и программистом контроллер или компьютер, который будет выполнять необходимые функции, разместить мотор и различные датчики с устройствами, которые требуются для правильной работы.

Итак, вам необходимо создать **четырёхколёсную модель малогабаритного робота, соответствующего следующим требованиям**:

- Размер робота не должен превышать 300 × 200 × 200 мм (д × ш × в);
- Корпус робота должен иметь возможность быть изготовленным путём 3D печати методом FDM (из пластикового прутка) или методом SLA (фотополимерная печать);
- Робот должен иметь возможность разворачиваться на месте;
- Всё оборудование, кроме датчиков, должно быть установлено внутри закрытого корпуса робота и не должно свободно перемещаться по нему;
- В качестве фиксирующих элементов можно использовать только защёлки, винты или саморезы;
- Для вывода проводов для подключения внешних устройств и датчиков должны быть предусмотрены соответствующие отверстия/пазы в корпусе;
- Датчики должны быть расположены вне корпуса таким образом, чтобы разрабатываемый вами робот имел возможность не врезаться в препятствия, расположенные спереди и сзади от него;
- Количество датчиков дистанции вы выбираете самостоятельно (в архиве дана 1 модель, но Вы можете её размножить);
- Все остальные устройства, 3D модели которых вам даны, должны быть установлены в Вашего робота.

В качестве ответа необходимо загрузить архив в формате `.rar`, `.zip` или `.7z`, содержащий в себе:

- Основной проект в файле приложения (в случае Fusion 360 — f3D);
- Сборку модели в формате `.step` или `.stp`;
- Скриншоты с различных сторон — спереди, сзади, справа, слева, сверху, снизу, а также под разными углами (до 10 шт.) в формате `.jpg` или `.png`;
- Скриншоты разобраной модели с указанием, как планируется собирать модель — до 20 шт. в формате `.jpg` или `.png`;
- К названию файла необходимо добавить материал и технологию производства, например, `*_fdm_PLA.step`;
- Все компоненты модели внутри архива должны иметь осмысленные названия (например, «колесо» или «крепление двигателя» или «винт крышки двигателя 1» и т. д.) и каждый элемент должен быть выполнен в виде отдельного компонента.

Если размер архива превышает 5 Mb, то загрузите его в облачное хранилище и вставьте ссылку в поле ответа. Не забудьте убедиться, что разрешили его просмотр и скачивание.

Для того, чтобы не начинать всё с нуля, вам доступны следующие элементы (3D модели):

- Контроллер Arduino UNO R3 (Arduino UNO R3.step, можно использовать 1 раз);
- Ультразвуковой датчик дистанции (Distance sensor hc-sr04.STEP, можно размножать бесконечно);
- Двигатель постоянного тока (DC motor.STEP, можно использовать до 2-х раз);
- Драйвер двигателя постоянного тока (DC motor driver.stp, можно использовать до 2-х раз);
- Батарея питания типа 18650 (18650 battery.STEP, можно использовать до 2-х раз);
- Держатель для батареи типа 18650 (18650 battery holder.stp, можно использовать до 2-х раз);
- Датчик утечки опасных газов MQ-2 (MQ-2.step, можно использовать до 2-х штук).

Рекомендации к выполнению:

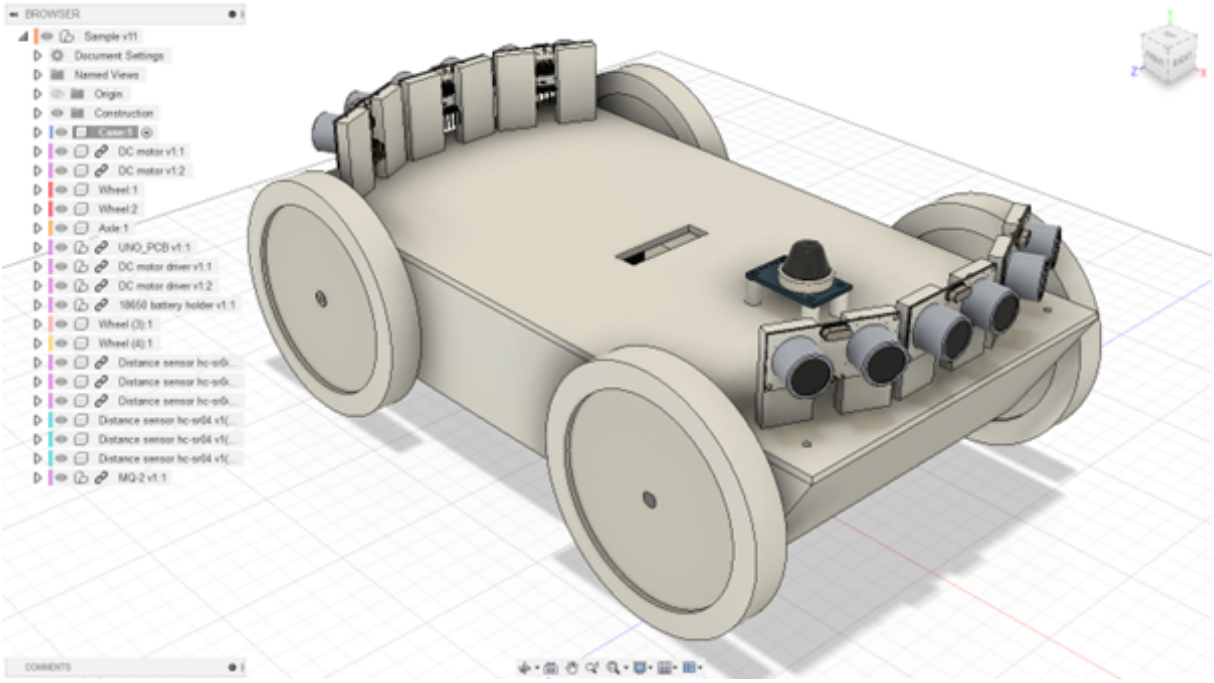
- Для винтов и саморезов отверстия должны быть меньше их диаметра, например, для винтов М3 диаметр отверстия лучше делать 2,75 мм, для М4 — 3,6 мм;
- Между деталями, которые планируется вставлять друг в друга старайтесь оставлять зазор в 0,5–1 мм для того, чтобы не возникло проблем при сборке;
- Старайтесь делать детали с небольшими наклонами с бОльшим количеством прямых линий для более успешной возможности печати.

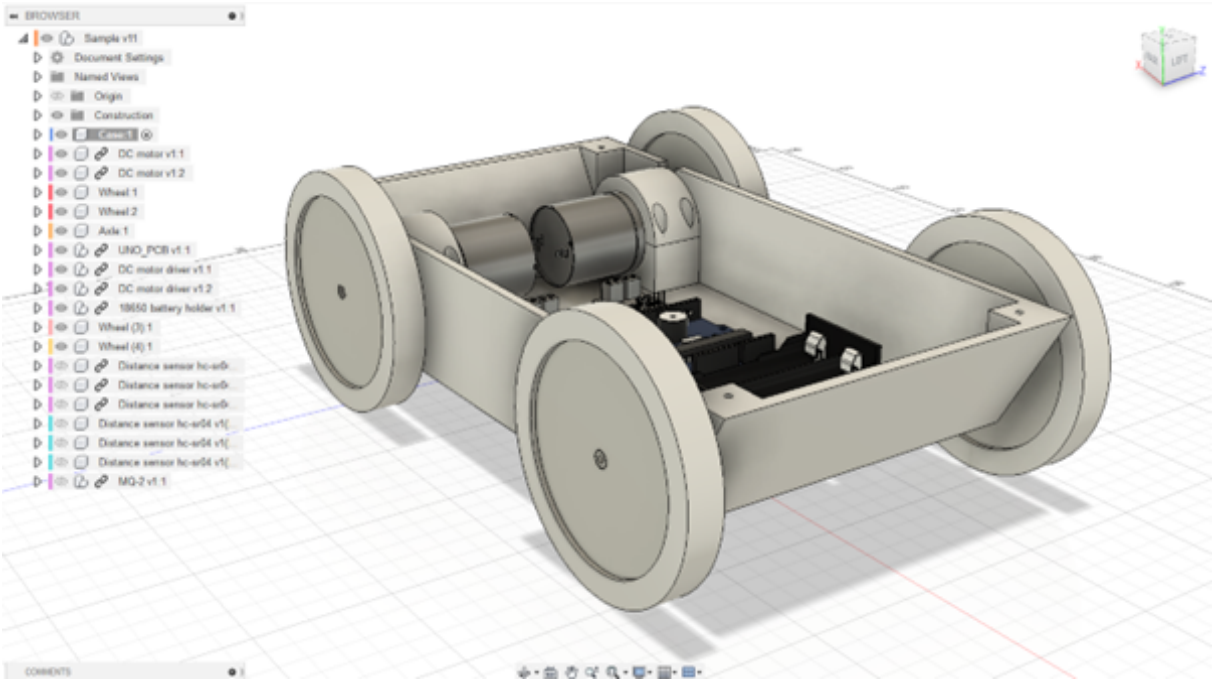
Критерии оценивания

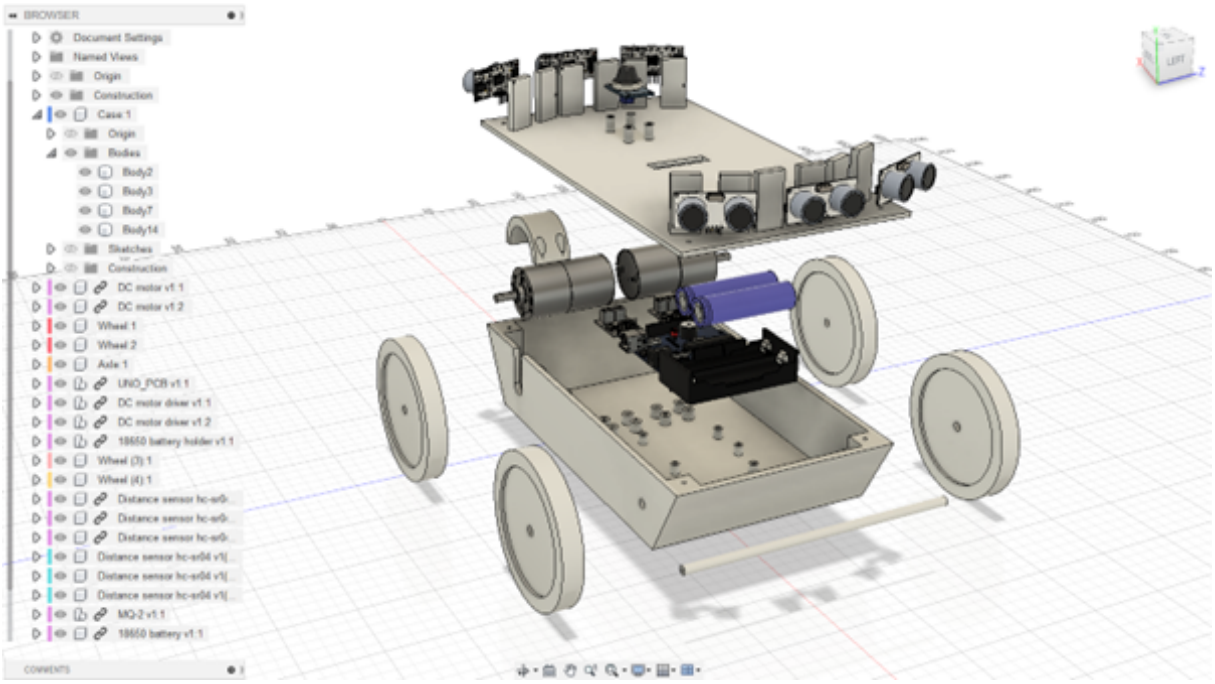
1. Реализуемость разработанной модели технологии изготовления — максимально 15 баллов;
2. Аккуратность и структурированность проекта — максимально 5 баллов;
3. Продуманность расстановки и закрепления компонентов — максимально 10 баллов;
4. Полнота снимков для представления модели — максимально 5 баллов;
5. Полнота снимков для представления сборки — максимально 10 баллов;
6. Полнота реализации — максимально 5 баллов.

Решение

Пример корректно выполненного задания







Схемотехника. Программирование Arduino

Задача IV.3.1. (50 баллов)

Темы: схемотехника, симуляция, Arduino, программирование, робототехника.

Условие

Итак, вы совместно (совместно ведь, правда?) создали прекрасный корпус с начинкой для вашего будущего робота, однако, думаю, не стоит говорить, что без органов чувств, которыми выступают сенсоры, без того, что выполняет команды, да без мозга всего вашего робота.

Обычно, при проектировании, выполняют разделение так называемого исполнительного (как и следует из названия — его задача — исполнять команды), тактического (в его задачу входит выбор тактики движения) и стратегического уровня (в его задачу входит быть «мозгом» и решать как действовать).

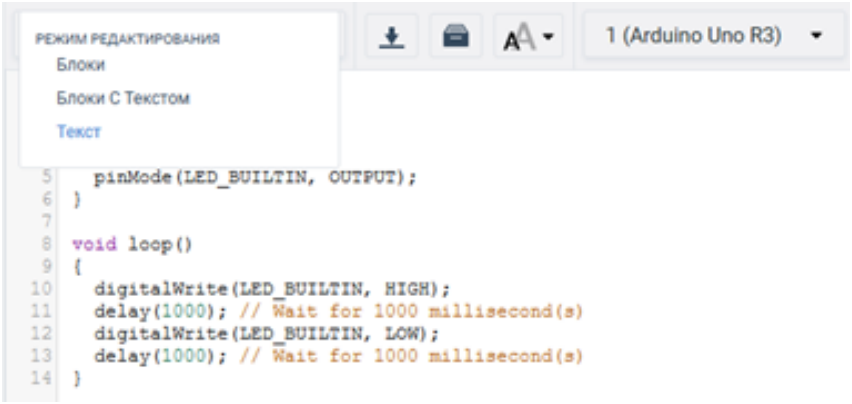
В рамках второго задания мы предлагаем вам разработать систему по сбору информации с датчиков, а также реакции на них в автоматическом режиме и обмену данными с верхним (стратегическим) уровнем.

Первая часть задачи состоит из сборки электрической схемы.

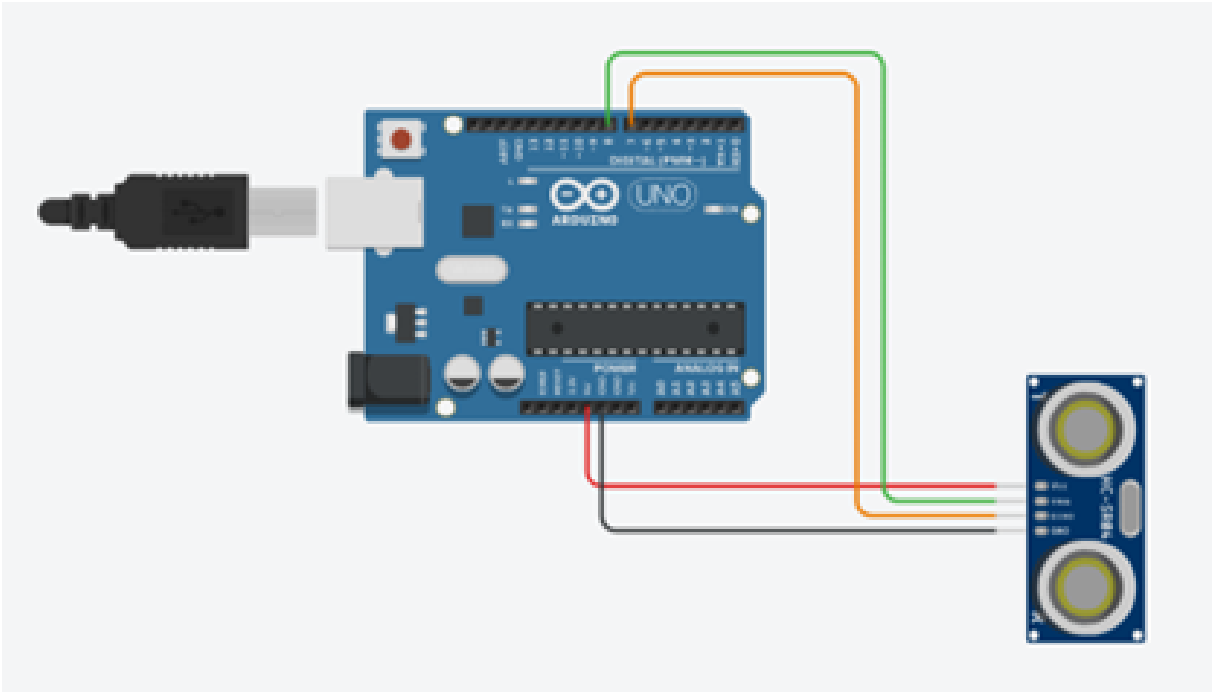
В среде разработки Autodesk tinkercad (https://www.tinkercad.com) необходимо разработать проект вашей системы управления, пользуясь следующими правилами:

- В качестве контроллера используется предоставляемый средой Arduino UNO R3;
- Написание кода должно осуществляться в текстовом формате, то есть запрещено использование программирования при помощи блоков (то есть в разделе

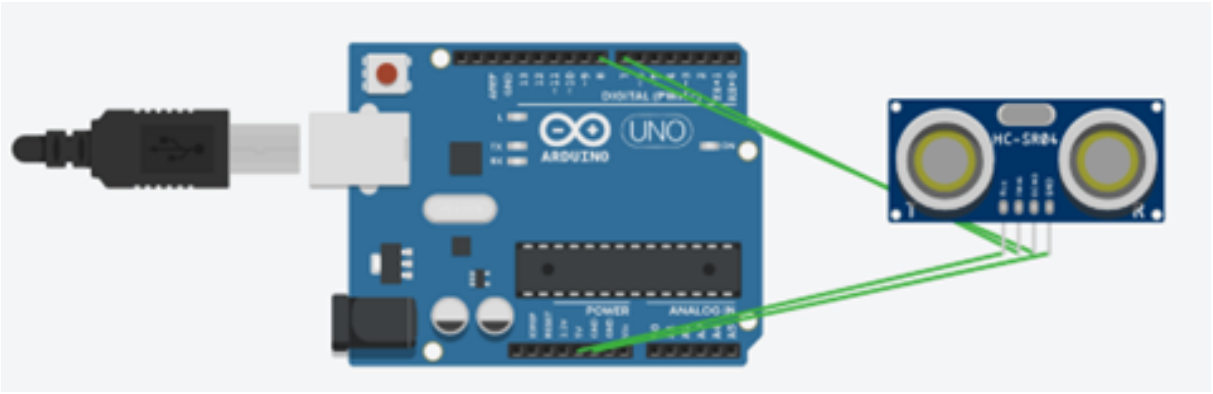
«Код» в правом верхнем углу должен быть выбран режим редактирования «Текст»);



- Количество датчиков дистанции должно совпадать с количеством выбранных вами датчиков на этапе конструирования;
- При соединении компонентов при помощи линий связи необходимо использовать одинаковые цвета для одних и тех же типов линий (например, красный — для питания, чёрный для заземления, зеленый — аналоговый сигнал и т. д.), при этом выбор цветов произвольный;
- Датчики, которые на разработанной вами модели стояли спереди должны на схеме располагаться справа, те, которые расположены сзади — слева, те, которые справа — внизу, те, которые слева — сверху;
- Аналогично — для двигателей — правый должен находиться внизу, левый — сверху, при этом движению колеса вперёд соответствует положительный показатель скорости на моторе, а движению назад — отрицательный;
- Расположение остальных компонентов может быть произвольным;
- Компоненты необходимо по-возможности соединять линиями связи максимально аккуратно. Рекомендуемый визуальный стиль — соединение «шинами»:

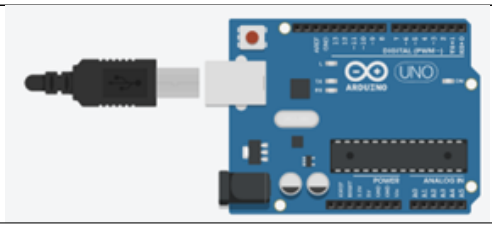


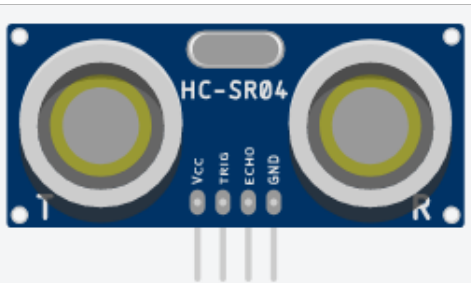
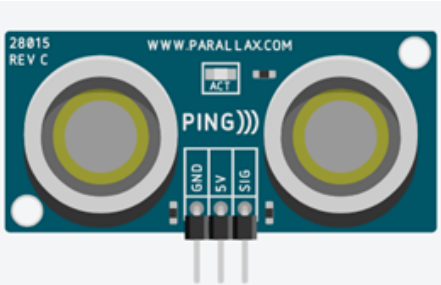
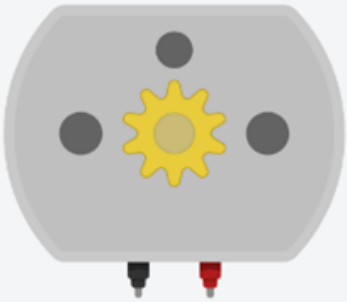
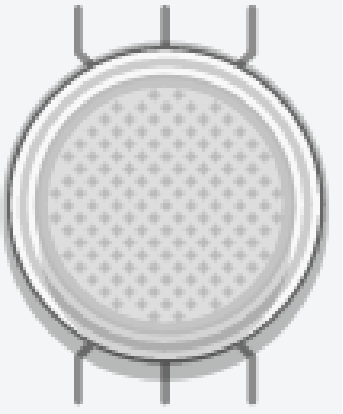
И старайтесь не делать так:



- Допускается использование макетных плат для облегчения соединений;
- Для большей наглядности просьба давать компонентам осознанные имена — например, правому двигателю — «правый двигатель», датчику дистанции «датчик передний центр» и т. д.
- В случае отсутствия необходимого количества пинов у встроенной в симулятор Arduino UNO допускается сокращение количества датчиков дистанции или замены их на датчики с 1 выводом:

При этом, принимая во внимание, что не все устройства из конструкторской задачи имеют полного дублёра в симуляторе, при решении задачи просьба пользоваться следующими соответствиями.

Контроллер Arduino UNO R3	

Ультразвуковой датчик дистанции	 или 
Двигатель постоянного тока	
Драйвер двигателя постоянного тока	
Батарея питания типа 18650 Держатель для батареи типа 18650 Считаются единым элементом	
Датчик утечки опасных газов MQ-2	

Вторая часть состоит из написания программного кода для Arduino Uno симулятора tinkercad.

Вам необходимо, для собранной вами схемы, разработать программу, которая будет выполнять следующие действия:

Собирать информацию со всех используемых датчиков дистанции а также датчика опасных газов и отправлять её в серийный порт в следующем формате: «ДД1;ДД2;ДД3;...ДДN; ДГ», то есть после одного опроса всех датчиков дистанции должна быть сформирована строка, в которой показания датчиков дистанции будут идти по порядку и разделяться точкой с запятой. После окончания одного опроса и вывода строки с показаниями датчиков дистанции, строчка с новыми данными уже следующего опроса должна начинаться с новой строки, например:

«ДД1;ДД2;ДД3;...ДДN; ДГ»
«ДД1;ДД2;ДД3;...ДДN; ДГ»
«ДД1;ДД2;ДД3;...ДДN; ДГ»
«ДД1;ДД2;ДД3;...ДДN; ДГ»
и т.д.

Где ДД1 — показания дистанции 1 датчика, ДД2 — 2 датчика, ДД3 — 3 датчика и так далее, ДГ — показания датчика газов.

Эти данные могут быть очень важны нашему стратегическому уровню для работы. Это одна первая часть программы.

Однако, этого мало, правда? Мало собирать данные, нам нужно ещё и как-то действовать, исходя из этих данных. Во второй части программы, по получаемым данным с датчиков дистанции, вам необходимо таким образом управлять вашими двигателями, чтобы робот имел возможность избежать столкновение с препятствием, которое приблизилось к роботу на расстояние меньше, чем 60 см. Каким образом это сделать — остаётся на ваше усмотрение.

В качестве примера можно привести следующий небольшой кусок алгоритма:

Допустим, мы имеем робота, у которого 3 датчика дистанции спереди — один по центру, один смотрит направо под углом 45 градусов и один — налево под углом также 45 градусов.

- Если на всех датчиках дистанции расстояния больше 600 мм — мы двигаемся прямо (на оба двигателя при помощи драйвера подаётся максимальное напряжение);
- Если показания правого датчика дистанции стали меньше 600 мм, то мы уменьшаем напряжение на левом двигателе;
- Дописать, вставить картинки.

Требования к оформлению программного кода:

- Старайтесь использовать в написании кода функции;
- К каждой написанной функции должны быть написаны комментарии на русском языке;
- Также комментариями следует пояснить действия вашей программы;
- Переменным, используемым в ходе написания программы следует давать осмысленные имена, также, как и функциям.

В качестве ответа необходимо приложить ссылку на проект в tinkercad, где будет собрана ваша электрическая схема, введён программный код. В комментариях к коду просьба в текстовом формате внести описание вашего алгоритма реакции на препятствия.

Критерии оценивания

1. Правильность составления электрической схемы — 15 баллов;
2. Аккуратность составления и читаемость электрической схемы — 10 баллов;
3. Правильность работы алгоритма избегания столкновений — 15 баллов;
4. Аккуратность и читаемость программного кода — 10 баллов;

Решение

Пример корректно выполненного задания

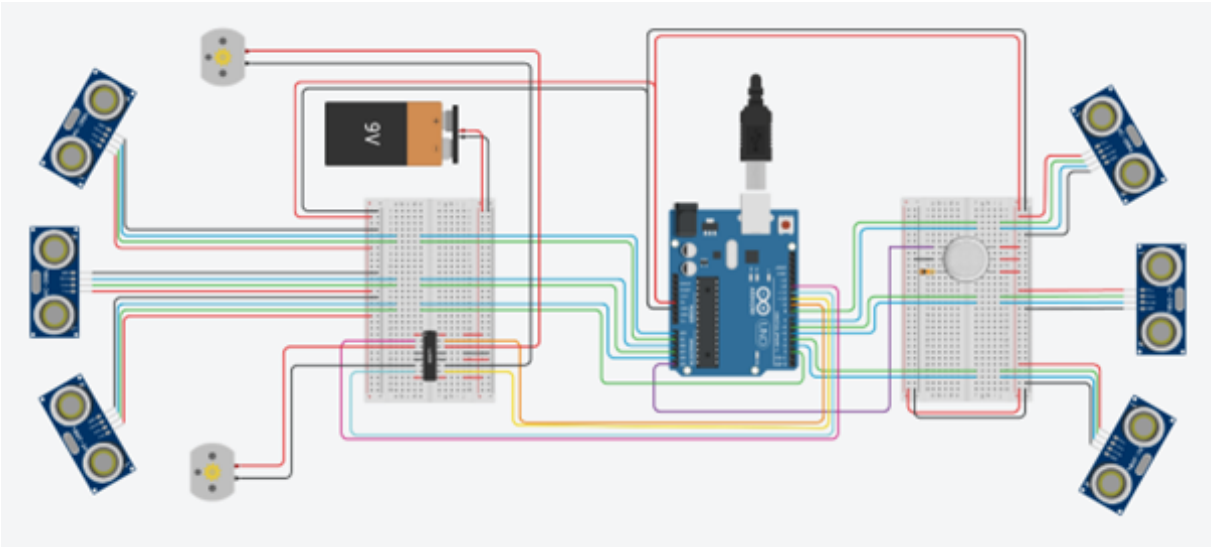


Рис. IV.3.1. Корректно собранная электрическая схема

Пример программы-решения

Ниже представлено решение на языке C++.

```
// Объявление пинов и их принадлежности
// Пины для датчиков дистанции
// Передние
#define FL_Trig 8
#define FL_Echo 7
#define FC_Trig 6
#define FC_Echo 5
#define FR_Trig 4
#define FR_Echo 3
// Задние
#define RL_Echo A0
#define RL_Trig A1
#define RC_Echo A2
#define RC_Trig A3
```

```
#define RR_Echo A4
#define RR_Trig 2
// Датчик газов
#define Gas_sensor A5
// Драйвер двигателей L293D
#define Left_DC_1 9
#define Left_DC_2 10
#define Right_DC_1 12
#define Right_DC_2 11

// Объявление переменных для ультразвуковых датчиков
long FL_US = 0;
long FC_US = 0;
long FR_US = 0;

long RL_US = 0;
long RC_US = 0;
long RR_US = 0;

// Объявление набора команд для управления двигателями

// Функция для чтения показаний датчика дистанции
long Read_US_Sensor(int Trigger_Pin, int Echo_Pin)
{
    digitalWrite(Trigger_Pin, LOW);
    delayMicroseconds(2);
    digitalWrite(Trigger_Pin, HIGH);
    delayMicroseconds(10);
    digitalWrite(Trigger_Pin, LOW);
    // Подсчёт дистанции в сантиметрах
    return 0.01723*pulseIn(Echo_Pin, HIGH);
}

// Функция считывания показаний всех датчиков дистанции и занесение их в
↪  переменные
void Read_ALL_US_Sensors()
{
    FL_US = Read_US_Sensor(FL_Trig, FL_Echo);
    FC_US = Read_US_Sensor(FC_Trig, FC_Echo);
    FR_US = Read_US_Sensor(FR_Trig, FR_Echo);

    RL_US = Read_US_Sensor(RL_Trig, RL_Echo);
    RC_US = Read_US_Sensor(RC_Trig, RC_Echo);
    RR_US = Read_US_Sensor(RR_Trig, RR_Echo);
}

// Функция для отправки данных всех датчиков в серийный порт
void Send_Sensor_Data()
{
    Serial.print(FL_US);
    Serial.print(";");
    Serial.print(FC_US);
    Serial.print(";");
    Serial.print(FR_US);
    Serial.print(";");
    Serial.print(RL_US);
    Serial.print(";");
    Serial.print(RC_US);
    Serial.print(";");
    Serial.print(RR_US);
```

```
74      Serial.print(";");
75      Serial.print(analogRead(Gas_sensor));
76      Serial.println(";");
77  }
78
79  // Команды: F - полный вперёд, B - полный назад, L - прямо налево, R - прямо
    ↪  направо, S - стоп, Q - назад налево, E - назад направо
80  bool Exec_Motor_CMD(char cmd)
81  {
82      switch (cmd)
83      {
84      case 'F':
85          digitalWrite(Left_DC_1, HIGH);
86          digitalWrite(Left_DC_2, LOW);
87          digitalWrite(Right_DC_1, HIGH);
88          digitalWrite(Right_DC_2, LOW);
89          Serial.println("Forward");
90          return true;
91          break;
92      case 'L':
93          digitalWrite(Left_DC_1, LOW);
94          digitalWrite(Left_DC_2, LOW);
95          digitalWrite(Right_DC_1, HIGH);
96          digitalWrite(Right_DC_2, LOW);
97          Serial.println("Left");
98          return true;
99          break;
100     case 'R':
101         digitalWrite(Left_DC_1, HIGH);
102         digitalWrite(Left_DC_2, LOW);
103         digitalWrite(Right_DC_1, LOW);
104         digitalWrite(Right_DC_2, LOW);
105         Serial.println("Right");
106         return true;
107         break;
108     case 'B':
109         digitalWrite(Left_DC_1, LOW);
110         digitalWrite(Left_DC_2, HIGH);
111         digitalWrite(Right_DC_1, LOW);
112         digitalWrite(Right_DC_2, HIGH);
113         Serial.println("Back");
114         return true;
115         break;
116     case 'Q':
117         digitalWrite(Left_DC_1, LOW);
118         digitalWrite(Left_DC_2, LOW);
119         digitalWrite(Right_DC_1, LOW);
120         digitalWrite(Right_DC_2, HIGH);
121         Serial.println("Back Left");
122         return true;
123         break;
124     case 'E':
125         digitalWrite(Left_DC_1, LOW);
126         digitalWrite(Left_DC_2, HIGH);
127         digitalWrite(Right_DC_1, LOW);
128         digitalWrite(Right_DC_2, LOW);
129         Serial.println("Back Right");
130         return true;
131         break;
132     case 'S':
```

```
133             digitalWrite(Left_DC_1, LOW);
134             digitalWrite(Left_DC_2, LOW);
135             digitalWrite(Right_DC_1, LOW);
136             digitalWrite(Right_DC_2, LOW);
137             Serial.println("Stop");
138             return true;
139             break;
140         default:
141             return false;
142             break;
143         }
144     }
145
146     void setup()
147     {
148         // Инициализация цифровых пинов на ввод и вывод
149         pinMode(FL_Trig, OUTPUT);
150         pinMode(FC_Trig, OUTPUT);
151         pinMode(FR_Trig, OUTPUT);
152         pinMode(RL_Trig, OUTPUT);
153         pinMode(RC_Trig, OUTPUT);
154         pinMode(RR_Trig, OUTPUT);
155
156         pinMode(FL_Echo, INPUT);
157         pinMode(FC_Echo, INPUT);
158         pinMode(FR_Echo, INPUT);
159         pinMode(RL_Echo, INPUT);
160         pinMode(RC_Echo, INPUT);
161         pinMode(RR_Echo, INPUT);
162
163         pinMode(Gas_sensor, INPUT);
164
165         pinMode(Left_DC_1, OUTPUT);
166         pinMode(Left_DC_2, OUTPUT);
167         pinMode(Right_DC_1, OUTPUT);
168         pinMode(Right_DC_2, OUTPUT);
169
170         Serial.begin(9600);
171
172         // Снимаем питание со всех двигателей (останавливаем их)
173         digitalWrite(Left_DC_1, LOW);
174         digitalWrite(Left_DC_2, LOW);
175         digitalWrite(Right_DC_1, LOW);
176         digitalWrite(Right_DC_2, LOW);
177     }
178
179     void loop()
180     {
181
182         Read_ALL_US_Sensors();
183         Send_Sensor_Data();
184         // Реакция на препятствия
185
186         if((FL_US < 60) && (FC_US < 60) && (FR_US < 60))
187         {
188             Exec_Motor_CMD('S');
189         }else if(RL_US < 60 && RC_US < 60 && RR_US < 60)
190         {
191             Exec_Motor_CMD('S');
192         } else if(((FR_US < 60) && (FC_US < 60)) || ((FL_US < 60) && (FC_US < 60)))
```

```
193     {
194         Exec_Motor_CMD('B');
195     }else if((RR_US < 60 && RC_US<60) || (RL_US < 60 && RC_US < 60))
196     {
197         Exec_Motor_CMD('F');
198     }else if(FL_US < 60)
199     {
200         Exec_Motor_CMD('R');
201     }else if(FR_US < 60)
202     {
203         Exec_Motor_CMD('L');
204     }else if(RL_US < 60)
205     {
206         Exec_Motor_CMD('Q');
207     }else if(RR_US < 60)
208     {
209         Exec_Motor_CMD('E');
210     }else{
211         Exec_Motor_CMD('F');
212     }
213 }
```

Высокоуровневое программирование

Задача IV.4.1. (50 баллов)

Темы: программирование, Python, карты, автоматизация, робототехника.

Условие

Дело осталось за малым! При поисково-спасательных работах является крайне важным найти оптимальный путь до нужной точки. Давайте научим нашего робота проходить любые лабиринты!

В этом задании вам требуется проложить маршрут по лабиринту, представленному в виде матрицы a (где 1 — это стены, а 0 — свободное пространство) от стартовой точки `start` до конечной `end`. При написании алгоритма следует учесть следующие ограничения:

- робот не умеет проходить сквозь стены;
- робот не имеет технологий телепортации;
- робот предпочитает экономить энергию и предпочитает ездить кратчайшим маршрутом.

Требования к оформлению программного кода:

- Старайтесь использовать в написании кода функции;
- К каждой написанной функции должны быть написаны комментарии на русском языке;
- Также комментариями следует пояснить действия вашей программы;
- Переменным, используемым в ходе написания программы следует давать осмысленные имена, также, как и функциям.

Примеры

Пример №1

Стандартный ввод

```
a (во всех тестах одинаковая) =
[[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
[1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1],
[1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
[1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1],
[1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1],
[1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1],
[1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1],
[1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
[1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
[1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1]];
start = 1, 1; end = 24, 5
```

Стандартный вывод

```
[(24, 5), (23, 5), (23, 6), (23, 7), (23, 8), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (23, 7), (23, 8), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (23, 7), (23, 8), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (23, 7), (23, 8), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (23, 7), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (23, 7), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
```

Стандартный вывод

```
[(24, 5), (23, 5), (23, 6), (23, 7), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (23, 7), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (23, 6), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (22, 5), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (22, 5), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (22, 5), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (10, 5), (9, 5), (9, 4), (8, 4), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 5), (23, 5), (22, 5), (22, 6), (22, 7), (22, 8), (21, 8), (20, 8),
(19, 8), (18, 8), (17, 8), (16, 8), (15, 8), (14, 8), (13, 8), (12, 8),
(11, 8), (10, 8), (10, 7), (10, 6), (9, 6), (9, 5), (9, 4), (9, 3), (8, 3),
(7, 3), (6, 3), (6, 2), (6, 1), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1)]
[(24, 15), (23, 15), (22, 15), (22, 14), (22, 13), (21, 13), (20, 13),
(19, 13), (18, 13), (17, 13), (16, 13), (15, 13), (14, 13), (13, 13),
(12, 13), (11, 13), (10, 13), (9, 13), (8, 13), (7, 13), (6, 13), (5, 13),
(4, 13), (3, 13), (2, 13), (2, 12), (2, 11), (2, 10), (2, 9), (2, 8),
(1, 8), (1, 7), (1, 6), (1, 5), (1, 4), (1, 3)]
```

Пример программы-решения

Ниже представлено решение на языке Python 3.

```
a = [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 1, 0, 0,
↪  0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 0, 1, 0, 1, 0, 0, 0, 0, 0,
↪  0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0,
↪  0, 0, 0, 0, 1], [1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1],
↪  [1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0], [1, 0, 0, 0, 1,
↪  1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1], [1, 1, 1, 0, 1, 1, 1, 1, 0, 1,
↪  0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0,
↪  0, 0, 0, 0, 1], [1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1],
↪  [1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 0, 1, 0, 1,
↪  1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 0, 1, 0, 0, 0, 0, 1, 0, 1,
↪  0, 0, 0, 0, 1, 0, 1, 1, 1, 1], [1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1,
↪  0, 0, 0, 0, 1], [1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1],
↪  [1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 1, 1, 0, 0,
↪  0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 0, 1, 0, 1, 0, 0, 0, 0, 0,
↪  0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0,
↪  0, 0, 0, 0, 1], [1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1],
↪  [1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1], [1, 0, 0, 0, 0,
↪  0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1], [1, 1, 1, 0, 0, 0, 0, 0, 0, 1,
↪  0, 0, 0, 0, 0, 0, 0, 0, 0, 1], [1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
↪  0, 0, 0, 0, 0], [1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1]];
start = 1,1
end = 24,5

def make_step(k):
  for i in range(len(m)):
    for j in range(len(m[i])):
      if m[i][j] == k:
        if i>0 and m[i-1][j] == 0 and a[i-1][j] == 0:
          m[i-1][j] = k + 1
        if j>0 and m[i][j-1] == 0 and a[i][j-1] == 0:
          m[i][j-1] = k + 1
        if i<len(m)-1 and m[i+1][j] == 0 and a[i+1][j] == 0:
          m[i+1][j] = k + 1
        if j<len(m[i])-1 and m[i][j+1] == 0 and a[i][j+1] == 0:
           m[i][j+1] = k + 1

def print_m(m):
    for i in range(len(m)):
        for j in range(len(m[i])):
            print( str(m[i][j]).ljust(2),end=' ')
        print()

m = []
for i in range(len(a)):
    m.append([])
    for j in range(len(a[i])):
        m[-1].append(0)
i,j = start
m[i][j] = 1

k = 0
while m[end[0]][end[1]] == 0:
    k += 1
    make_step(k)
    draw_matrix(a, m)

print(the_path)
```