Faculty of Engineering

Final Design Report of a Robotic Blackjack Dealer

A report prepared for the Department of Mechanical and Mechatronics Engineering,
University of Waterloo,
Waterloo, Ontario

by

Abdulmalik Ibrahim : 20580858
Gerard Salisi : 20607908
Kelvin Tezinde : 20619069
Nikhar Dhingra : 20601153

December 3, 2015.

9 + 10 Inc.
687 Tunnel Ave.
Waterloo, Ontario
K4N O2P

Dec 3, 2015

Professor William Melek
Director, Mechatronics Engineering
Department of Mechanical and Mechatronics Engineering
University of Waterloo
Waterloo, Ontario

Dear Professor Melek,

We are pleased to submit the enclosed design report for our final project in MTE 100.

This report examines the solution to the automation of the dealer's process in a game of blackjack. This report looks at the the design and implementation of such a system and discusses the software used in it’s operation.

The software data used in this project can be found included in the appendix.

If you have any concerns regarding the project, do not hesitate to call.

This report was prepared entirely on our own and has not received any previous academic credit at this or any other academic institution. We would like to acknowledge the MTE 100 TA’s in providing assistance to us in writing this report.

Sincerely,

Abdulmalik Ibrahim Gerrard Salisi Kelvin Tezinde Nikhar Dhingra

Table of Contents

List of Figures ii

Summary iii

1.0 Introduction 1

1.1 Design problem definition 1

1.2 Goals and Objectives 2

1.3 Constraints 2

1.4 Criteria 3

2.0 Mechanical Design and Implementation 4

2.1 Shuffler Design 1

2.2 Dealer Design 1

2.3 Rotating Base 1

3.0 Software Design and Implementation 8

3.1 Movement and Mechanics 7

3.2 Logistics Operations 7

3.3 Testing Procedures 7

4.0 Project Management 6

5.0 Conclusion and Recommendation 6

5.1 Conclusion 7

5.2 Recommendations 7

5.2.1 Mechanical recommendations 9

5.2.2 Software recommendations 9

6.0 References 10

Appendix A: Software Program 11

List of Figures

Figure 1: The BlackJack Dealer ……...……..1
Figure 2: The Shuffler ……...…….1
Figure 3: The Dealer ……..…….……1
Figure 4: The Rotating base …….,……...……1
Figure 5: Gantt chart for task deadline distribution…...……...……...1

Summary

This report goes in depth into describing a solution for a problem commonly faced by casinos in the present day. Casino's spend a lot of money hiring professionals to play as a dealer in a game of blackjack. The dealer's process, being highly repetitive, and very simple, can easily be automated and implemented by a robotic system. This report explores and explains the design and implementation of a robotic system set out to replace the dealer in a game of blackjack.

A four man team of students are primarily responsible for the design and implementation of this system. It's implementation followed a four man distribution of tasks and a guided timeline of events leading to the deadline of the project. The design of this system is strictly based on the constraints and criteria outlined by the company requesting the system. The constraints to be met are: must be a table-top system and must automate the dealer's process

The system's physical design is unique and dependent on three crucial components, of which are: the shuffler component, the dealer component and the rotating base, all of which define the dealer's role in a game of blackjack. The software program implemented in the system coordinates the movements of the robotic system during the game, as well a record keeping of wins during gameplay and a winner's declaration after gameplay, similar to a blackjack dealer.

Testing of the physical side of the system was mainly observational and followed rigorous revisions over the project's period of time. The test environment for the software was based on testing the functionality of different functions in the program, whilst recording and observing the output by the system.

A few recommendations on what could be done better, for both the physical and software components have been documented and listed on page (number) of this document. Some of these recommendations include but are not limited to: better time management, availability of more parts, better integration of software components e.t.c. Following the success of the project, the blackjack dealer can now be replicated and reproduced on a mass scale.

1.0 Introduction

1.1 Design Problem Definition

The design and implementation of a tabletop system that automates the dealer's process in a game of blackjack

1.2 Goals and Objectives

The goal is to successfully build a tabletop system to automate the dealer's process in a game of blackjack. The system should be able to shuffle, rotate and deal cards to players when appropriate. The system should also be able to keep score of players winnings as well as declare winners at the end of a game.

1.3 Constraints

In order for the robot to perform the problem definition it should meet the following constraints :

- The robot must shuffle cards
- The robot must deal cards
- The robot must wait for a button pushing before dealing cards
- The robot must be less than 3.56 x 2.56 meters in dimenscldions
- The robot must weigh less than 20 pounds

The constraints listed above remain consistent with the constraints listed in the past reports and the robotic system satisfies all of the constraints listed above. The constraints were met through reiterative tests and redesigns of components until the system had performed to a desirable output. The most difficult constraint to work with was the shuffler component. The mechanics of the shuffler component were not completely known to the team and so a design had to be made out from scratch. However, once the shuffler component was finished, the rest of the constraints followed through from it.

1.4 Criteria

The following are the criteria that the system is to be evaluated based on:

- The robot should be built entirely out of LEGO pieces
- The robot should be aesthetically pleasing
- The robot should weigh less than 15 pounds

2.0 Mechanical Design and Implementation

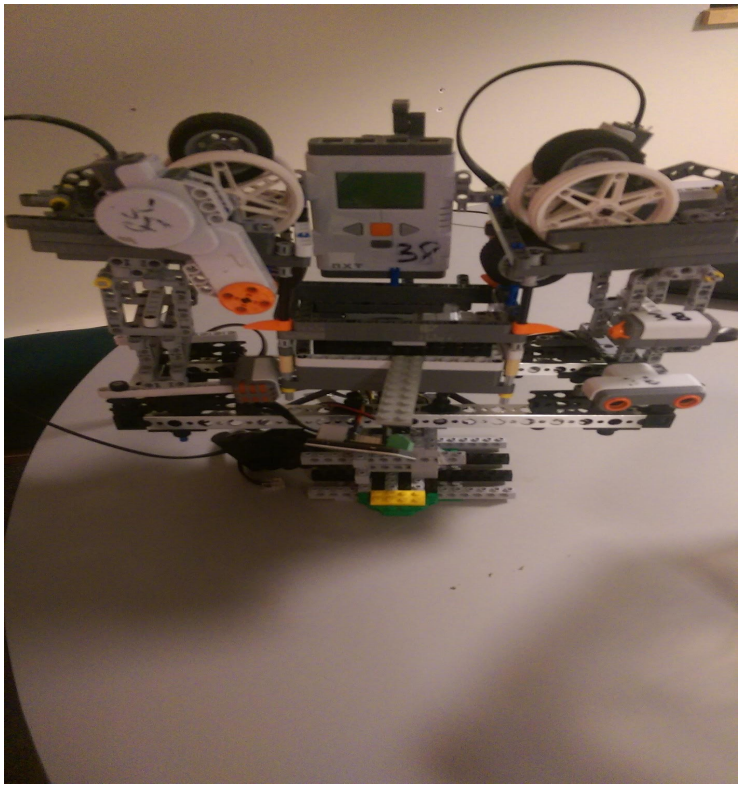

Figure 1.0: BlackJack Dealer

The mechanical design of the system can be can be split into the following three different components:

- The shuffler design
- The dealer design
- The rotating base design

2.1 Shuffler Design

Perhaps one of the biggest challenges of the whole project is the mechanical design of the shuffler. The preliminary design of the shuffler is to have two elevated platforms opposite of each other, with a lower base in between them. The platforms are to hold each half of a cut deck of playing cards. The design makes use of wheels, attached to the motors, at the bottom of each base that would throw the cards off the platform to the base, one by one, randomly. The small number of moving parts and the simplicity of the design are the factors that made this design our choice. Less moving parts means more reliable and easier debugging.

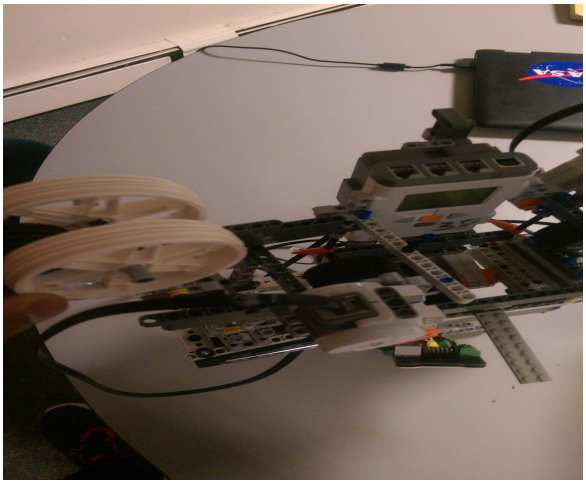

Figure 1.0: The Shuffler

The initial design has a major flaw which is later improved on during the building stage. The major flaw of the preliminary design is that the cards would often stick together and fall down to the platform together, this varies between two or three cards or to even the whole half of the cards in place. As a result of some testing and debugging, three additions/redesign are set in place to fix the flaw. Firstly, the wheel is set such that placing the cards on the wheel slightly inclines the cards to help move the cards more easily(See figure 1.1)

Secondly, a heavy arm consisting of spare rims and tires are connected to the end of the base as a weight on top of the cards to help the wheel catch on to the cards. (See figure 1.1) Lastly, a slot small enough for one or two cards to slide through is put in front of the platform to help the

number of cards that fall through to be consistent. These three additions fixed the flaw involved in our initial design and thus kept our initial and final design fairly similar.

The trade-off with this design is the amount of parts it needs. The two platforms and the lowered base needs a handful amount of materials for it to be stable enough to hold the cards in place. The slot also needs to be precise to only let a few cards off the platform go out and thus a handful of parts are put in place to ensure precision.

2.2 Dealer Design

The dealer design borrows a lot from the shufflers design. The mechanism is found in the base between the two platforms stated in the previous section. The dealer is similar to the shuffling mechanism where a wheel attached to a motor is placed at the bottom of the base to take care of the dealing. The slot is also implemented in this design however in comparison to the shuffling mechanism, there is no wiggle room in terms of the cards dealt. The goal is to have only two cards be dealt at the first rotation of the robot and then one at a time right after.

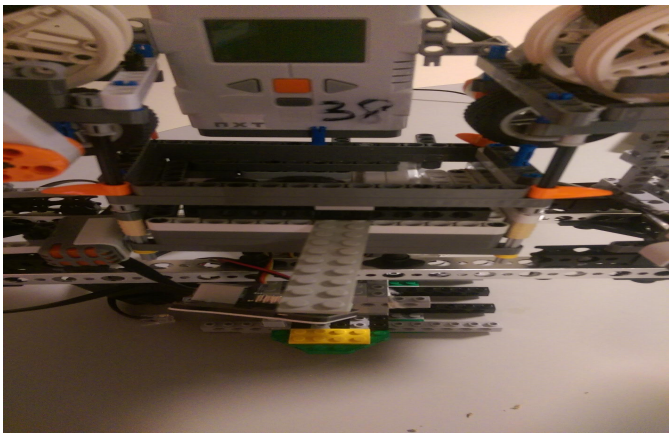

Figure 2.0: The Dealer

The preliminary design includes everything stated above plus a makeshift slide where the cards will get off from and slide unto player. However, due to the lack of time and resources, the slide is scrapped and instead the cards are simply thrown out slightly and the player gets it from the robot.(maybe an illustration or picture for clarity) The removal of the slide is not detrimental to the robot but instead helped with the overall effectiveness of the dealer mechanism. Having to slide off exactly one or two cards proved to be difficult. The timing of the motors, the slot size

and how the cards are shuffled into the base all factor in the effectiveness of the dealing. The redesign decreased the chances for error in terms of the number of cards dealt.

However, the chance for error is still present and thus is one of the trade-offs of this design. Through testing it is determined that the biggest factor in the effectiveness of the dealer is how the cards are placed. Therefore, in order to reduce the chance for error in the dealing, the cards will have to be fixed after the dealing part of the program. This, on top of the removal of the slide, introduces another trade-off which is loss of autonomy. The purpose of this robot is to automate dealing and this design slightly defeats that purpose. If more time is allowed to work on the robot, a complete redesign of this mechanism can take place or just work on some additions and improvements on the design.

2.3 Rotating Base Design

The rotating base is the final component to the Blackjack dealer. It's preliminary design differs vastly from the final design in various ways. During the early stages of the design, the group had drawn up a sketch that involves two poles supporting the system from both ends with a shaft connecting these two poles. A motor is then placed at the bottom and connects to the shaft, such that when the motor turned, the shaft also turned thus rotating the robot. However, as progress was made with the blackjack dealer, parts were being used up. The group foresaw a scarcity of parts as a potential problem for the rotating base and decided on its redesign to make use of a much more limited number of parts.

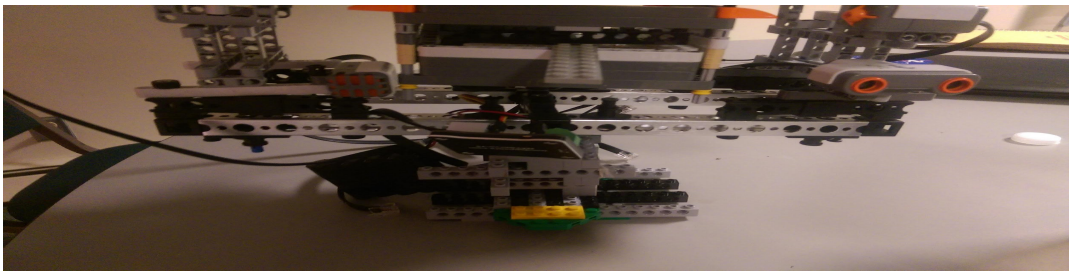

Figure 3.0: Rotating Base

The redesign of the rotating base is very similar to the initial design, but much more conservative with parts. The shufflers on either side of the blackjack dealer are supported by a stand (See image 1.0). A platform is then used to connect these stands and then a motor is placed beneath the platform to rotate the platform, thus rotating the system. This design, like the initial design, needed a fourth motor to operate and as such, a request for a multiplexer and extra motor was put out. Unfortunately, due to the limitation of parts, the group was unable to secure a multiplexer but was given the option of working with the TETRIX PRIME kit instead.

Another mechanical redesign of the rotating base was put in motion after realizing the group could no longer use Lego NXT parts for the base. Much of the second redesign of the part is kept the same, with the exception that the tetrix prime motor kit is to be used as opposed to the Lego NXT kit. The platform is now made from the stronger sturdier tetrix prime parts as opposed to the Lego NXT parts. The stronger platform is important in ensuring full and undeterred support for the system. The shuffler supports are anchored down to the new platform and the platform is attached to the servomotor, which when put in a vertical position allows for a rotary movement.

Around the servomotor are Lego blocks. These block serve to anchor the servo motor down to the ground and well as eliminate any instability where the servomotor is connected to the platform. The base is designed to give little no to room for the servomotor to move. However, based on this design a problem presents itself. A wire leading from the servo chip to the servo-motor wraps itself around the base while the motor is in operation. A fix to this is to simply have the servo chip and wire on level ground with the base to prevent it from wrapping around. Unfortunately, another wire connecting the NXT brick to the servo chip is present, and due to the position of the NXT brick higher up than the rotating base, the wire extends all the way down to the bottom if the servo chip is placed on level ground. This in turn wraps itself around the base instead.
The fix only presents another problem. A redesign of the software program such that the robot, once rotated in a full circle, will then rotate in the counterclockwise direction to unwrap the wire

below addresses this problem. The trade off in this case is the seen wire wrapped around the base, which slightly alters the robot's appearance.

3.0 Software Design and Implementation

The design of the software used to operate the blackjack dealer can be broken down into the two following sections:

- Movement and Mechanics
- Logistics

3.1 Movement and Mechanics

For the first part of its operation, the program prompts the user for the number of rounds to be played through a function call. This function is named count and returns the count value input by the user. The user can then input the number on the NXT brick made possible with the use of buttons and the screen. The reason for this carries into the logistics branch of the code. The blackjack dealer is able to announce a winner after a certain number of rounds which truly adheres to the concept of automation in the dealer's process. Once the user inputs the number of rounds to be played, the program stores and uses that information in a for -loop to determine the amount of times the game will be played.

Within that for loop lies the intricate parts of how the game is played. A shuffle function is called and that initiates the shuffling operation. Once this is done with, following immediately after is the dealing of the cards. The rotate and deal function is called for this operation. The rotate and deal function calls upon the rotate function, which rotates the base until the ultrasonic sensor detects an individual, as well as the deal function, which simply deals a card. The rotate and deal function is placed within a while loop who’s stopping condition is a full 360 degree turn measured by the motor encoder.

Within the while loop is a counter that increments each time a player is detected to determine the number of players sitting across from the table. After storing the number of players, the hit

or pass function is called immediately afterwards and placed within a for loop who's stopping condition is the number of players earlier recorded. The hit or pass function waits for a user to clap and get dealt a card or hit the touch sensor to pass. A timer is also used within the function to give the player 36 seconds to make these choices otherwise the robot will assume a pass.

The rotate function is called beneath the hit or pass function such that the robot rotates to the next person and call the hit or pass function again until it has done so for all persons. Once this is done, the robot commences on the second part of the program

3.2 Logistics Operations

The logistics operations require that the user inputs the winner of a certain round. The program stores a point to the winner in an array of persons. Once all rounds are completed and all winners have been given their points, a comparison is made below the for loop to compare player points and find the player with max number of winnings. Immediately following that is another comparison operation that checks to see if there is a tie.

If a tie exists is present, the robot outputs a message to the screen indicating such and is automatically set the number of rounds to just one. The winner of the round wins the tie and thus the game. If no tie is present, the robot outputs a message to the screen declaring the winner of the rounds. It then calls the reset function that prompts the user to end the program or reset it. The reset function takes in a reset value and that value changes depending on the action performed by the user.

3.3 Testing Procedures

The programs operation heavily rely on the functionality of the functions and as such, as soon as the functions were written, they were tested. The test environment for the functions involves testing one function at a time and observing the output on the system. Once, the functions are tested and in the clear, the main program is written and it undergoes its own sets of tests as well.

The Mechanics and Movements portion of the program is first tested by commenting out the logistics portion of the program. In essence, only the for loop which houses the code for the mechanics of the game remains. Within that for loop, are two loops which are also tested independently. Similar to the functions above, the testing procedure here involves running only the desired loop and making observations on the output through the system. Once all the loops within the for-loop are tested, debugged and working, the entire for loop was made operational and was tested as well. The outcome of the test was desirable.

The logistics part of the program was tested as well, following the same testing procedures as is outlined above. The only difference here being that the output is seen and recorded on the NXT brick's screen as opposed to observing a physical outcome by the robot. Once the logistics branch is tested and debugged, the mechanics and movements merge with the logistics in main to form a complete program. At this stage, no further tests are conducted. The functionality of the program depends on the functionality of the two independent branches.

4.0 Project Management

The workload of Lego project divided among the teammates in an orderly manner. Kelvin Andre Tezinde and Abdulmalik Ibrahim are working on developing the robot's software program. Gerard Salisi took the initiative in conceptualising the mechanical design of the robot and is being aided by the rest of the team. Nikhar Dhingra is the head of this project and it is her duty to make sure that the task deadlines are met by her team. Initially the team proposed various task deadlines as shown below in the Gantt chart.

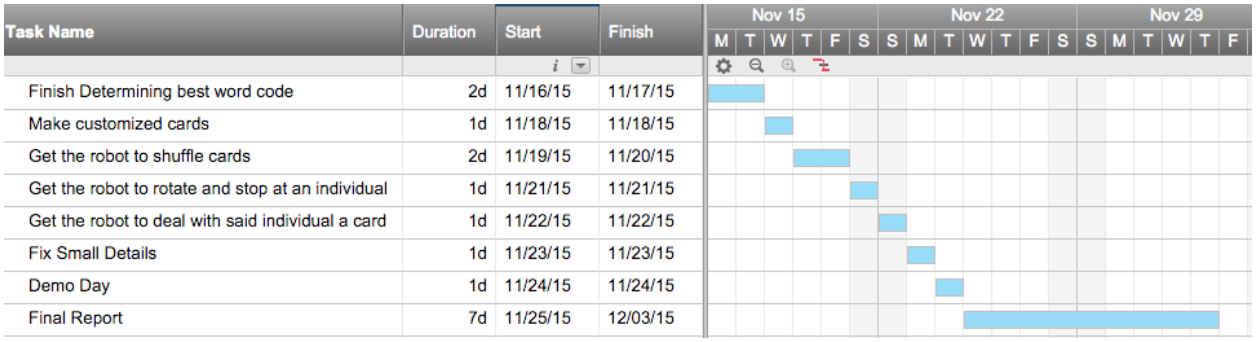

Figure 4.0: Gantt Chart for the task deadline distribution

The team has made some stupendous achievements in meeting with deadlines of software design, however, getting some aspects of the mechanical design working has been a cause of worry to the team till the Demonstration Day. The card dispenser slot of the robot , whose final prototype was due by 22nd November 2015, was finished on 23rd November 2015. The base of the robot was also fixed in the early morning of 24th November 2015. The team faced many hurdles in ensuring the stability of the card slots of the robot, however, after major brainstorming sessions and minor fixations of the robot, the team members showcased a successful presentation of the robot's operation on the Demo Day.

5.0 Conclusions and Recommendations

5.1 Conclusion

The group has successfully designed and implemented a working system to automate the dealer's process in a game of blackjack, thus meeting the objective of this project. The mechanical design is unique, satisfying the constraints laid out as well as achieving some of the criteria outlined. In addition to having satisfied the criteria and constraints, it should also be noted that some additional features (the logistics branch) were implemented in the software to provide more functionality and true automation to the system.

To recap on some of the design features, the shuffler component on the mechanical side allows for easy shuffling of cards as would a dealer in the game of blackjack. The dealer takes on the

responsibility of the working hand of a blackjack dealer and deals players their cards, whilst working hand in hand with the base which rotates the system to a new player. The software component integrates two branches, one which primarily deals with the movements of the system and the other, the logistics of the game.

5.2 Recommendations

The recommendations for improvement in mechanical and software design are described in the following sections.

5.2.1 Mechanical Recommendations

Dealer:

The effectiveness of the dealer varied due to several factors (such as the size of the slot, the grip of friction between the wheel and cards, etc). Because of this entropy, the dealer would sometimes deal two cards, or one, or none when it was required to deal one or two. A mechanical recommendation regarding the dealer could include:

- Instead of using lego parts for the slot (as they tend to be flimsy and in consequence, make the slot hole move), a 3D printed alternative should have been looked at. This is because the 3D printed slot would have been much more firm, thus making the effect of the motors on it diminish, allowing for the slot to remain unchanged.

Location of the ultrasonic sensor:

Due to the location of the ultrasonic sensor, the robot detected undesired objects then stopped and dealt cards to said objects. Because of this, the inherent function of the robot was slightly deferred (as once it detects and object, there is a wait, meaning it could potentially skip a person).

- If the location of the ultrasonic sensor was more centralized than such an issue would be diminished (but not completely eliminated). A good area to place the sensor would be on top of the brick.

5.2.2 Software Recommendations

The program for our robot is fairly simple and the only recommendation we have is to add in another function that unwinds the wire from the base before moving on with the rest of the program. Other than this, we can optimize the program to increase efficiency.

Appendix A: Software program

```
#include "NXTServo-lib-UW.c"

void shuffle()
    {
        motor[motorA] = -45;
        wait1Msec(100);
        motor[motorC] = -45;
        while(nNxtButtonPressed!=3);
        motor[motorA]=0;
        motor[motorC]=0;

    }


void deal(int numCards)
    {
        motor[motorB] = -45;
        wait1Msec(numCards * 500);  // takes 0.5 seconds to deal one card
        motor[motorB]=0;
    }

 void rotate()
    {
          setServoSpeed(S3, 1, 10, -17, 12);
          motor[motorA] = 15; //See note at the bottom as to why motor[motorA] is initialized
       wait1Msec(500);     //to get past already detected player
             while (SensorValue[S1] > 100);
        motor[motorA] = 0;
          setServoSpeed(S3, 2, 0, -12, 16);
    }


void rotate_and_deal()
    {
        rotate();
        deal(2);
```

```
    }


void hit_or_pass()
{
        time1[T1] = 0;

        while ( time1[t1] < 36000 && SensorValue[s2] == 0)
        {
                if (sensorValue[s4] > 70)
                {
                        deal (1);
                        wait1Msec (2000);
                }
        }
}


int countt ()  // Counting mechanism
{
        int count = 0;
        while (nNxtButtonPressed != 3)
        {
                if (nNxtButtonPressed == 1)
                {
                        count ++;
                        while (nNxtButtonPressed ==1);
                        displayString(1, "Number: ");
                        displayString(2, "%d", count);
                        while (nNxtButtonPressed == -1);
                }

                else if(nNxtButtonPressed == 2)
                {
                        count --;
                        while(nNxtButtonPressed == 2);
                        displayString(1, "Number: ");
                        displayString(2,  "%d", count);
                        while (nNxtButtonPressed == -1);
                }
        }
        eraseDisplay();
        return count;
}


 void reset (int &reset_value)
      {
          NxtDisplayString (0, "Hit orange: END");
          NxtDisplayString (1, "Hit touch: RESET");
```

```
        while (nNxtButtonPressed!=3 && SensorValue[S2] != 1);

        if (SensorValue[S2] == 1)
            reset_value = 1;

        else if (nNxtButtonPressed == 3)
            reset_value = 0;
    }



task main()
{

    int reset_value = 1;

       SensorType[S1]=sensorSONAR;
       SensorType[S2]=sensorTouch;
       SensorType[S3] = sensorI2CCustom9V;
      SensorType[S4]= sensorSoundDB;

    while (reset_value == 1 || reset_value == 2)
        {
            int max_score = 0, n = 0, no_players = 3, winner = 0, rounds = 1, c = 0;

            int person[10] = {0, 0, 0, 0, 0, 0 , 0, 0, 0, 0}; // Robot C did not allow
initialization in the form {0};

            if (reset_value == 1)
                {
                    nxtDisplayString (1, "Right to increase");
                    nxtDisplayString (2, "Left to decrease");
                    wait10Msec(200);
                    eraseDisplay();
                    nxtDisplayString (0, "Number of rounds" )
                    rounds = countt();
                }

            for (int i = 1; i<=rounds; i++)
                {

                    shuffle ();

                    while (nNxtButtonPressed == 3);
                    while (nNxtButtonPressed !=3);


                    nMotorEncoder[motorA] = 0;
```

```
                    while (nMotorEncoder[motorA] < 780) // 780 rotates 360 as measured by
motorEncoder[motorA]
                        {
                            rotate_and_deal ();
                            no_players++
                            wait10Msec(200);  //Waits for the player to pick up a card
                        }




  for (int n = 0; n < no_players; n++)
                        {
                            hit_or_pass ();
                            rotate ();
                        }

                    displayString (0, "Input a winner");
                    while (nNxtButtonPressed == 3);
                    winner = countt();

                    person[winner]++;

                }

            reset_value = 1;

            for (int i = 1; i <= no_players; i++) //Checks for max number of wins
                 {
                    if (person[i] > max_score)
                    {
                        max_score = person[i];
                        c = i;
                    }
                 }

        int counter = 0;

             for (int i = 1; i <= no_players; i++) // checks to see if there is a tie
               if (person[i] == max_score)
                      counter ++;


               if (counter >= 2)
                {
                    reset_value = 2;
                    nxtDisplayString (0, "Tie");
                    nxtDisplayString (1, "Will play on more round");
                    nxtDisplayString (2, "Hit button to continue");
```

```
                while (nNxtButtonPressed == 3);
                while(nNxtButtonPressed == -1);
                eraseDisplay();

            }

        else
            {
                nxtDisplayString (0, "Winner is person" "%d", c);
                nxtDisplayString (1, "Max score = %d", max_score);
                nxtDisplayString (2, "Hit button to continue");
                while (nNxtButtonPressed == 3);
                while(nNxtButtonPressed == -1);
                eraseDisplay ();
            }


    while (nNxtButtonPressed == 3);

    if (reset_value != 2)
        reset(reset_value);

}


}
/* Note to Marker, the reason why we initialize Motor[motorA] = 15; is because the servo
motors do not measure motor encoder values and so we have to use the Encoder value for
motor[motorA] to get a measure of the encoder count.*/
```