**Lab6.Ogbondah**

**LAB 6**

**SECTION D**

**Chimzim Ogbondah**

**SUBMISSION DATE: 10.16.2018**

**DATE: 10.16.2018**

# Problem

1. The purpose of this lab was to be able to understand and use an integer array along with prior knowledge of C to create a working program of the game Bop it. (other knowledge includes things like functions, branches (if else statements, while loops and for loops) and know which libraries to import for our own benefit. Ex is stdlib.h for the rand function)

# Analysis

1. The problem for this lab was to create a game that acted like the Bop it game where the user would be prompted to press a button to begin the game. From there a random button would be printed to the screen letting the user know which button to press and like in the game there is a timer and so the user only has a certain amount of time to answer correctly. If the user answered in correctly or ran out of time the game was to be exited printing out how many rounds the user made it past.

# Design

1. Our problem was to create a Bop it game from scratch meeting all constraints from the lab instructions and prove that it worked correctly through a demo. I used a step by step process for solving this problem. First by gaining small wins that would eventually sum up to the completed game solving the problem. (Note that TRUE took on the value of 1 and FALSE took on the value of 0)
   1. I first broke up the game into different areas (time check, button check, start menu etc.)
   2. Next, I looked at the different areas and made functions
   3. For the time check I set the pervious time variable 0 and then created an if else statement that checked to see if the user was within the given time limit. If they were the previous time was saved as the current time, and from then on the previous time and the current time where added together returning true if it was still within, if it was no longer within it returned false. (will take on the scanned in time for the drsd function)
   4. Then I created the int array which was based off how the controller scanned in the button values. I then assigned numbers 1-4 for Triangle – Square
   5. After this I worked on the button function where I created and if else statement the checked which button had been pressed. I created variables Triangle1- Square4 that held the values of that matched with Triangle – Square. Inside the if else I nested another if else that checked if the desired button had been pressed by checking if Triangle1- square4 == the int array with the rand function. If it was true it returned True else, it returned false. (will take on the scanned in buttons from the controller)
   6. Then I created the start menu button where if the circle button was pressed it returned true, if not it returned false.
   7. Finally, I created one more function that took on the rand button and using an If else statement it checked to see if it was equal to Triangle1 – Sqaure4 and then printed out the correct button to the terminal for the user to press.
   8. Once I had all my function created I went to the main function where I used 3 while loops
      1. The first while loop just counted from zero and waited for the user to press the circle button (while start\_menu(circle) == 0)
      2. The next while loop started the game by setting game = 1 and then exiting the loop
      3. Finally the third one took on the variable game (while game ==1) and ran the code of the game printing out the random button then printing the time and using an if else statement checked to see if the user was right and within the time frame(the round would go up by one) if the user was incorrect with the button it would print out sorry you have answered wrong and then print out the score. Finally, if the time ran out it would print out sorry you have run out of time regardless of whether the correct button was pressed. If the user answered incorrectly or ran out of time the If else statement would be exited, which would then print out how many rounds the user made it to, and finally ending the while loop by returning the value of zero.
   9. After my logic for the game was written I added the stdlib.h, and stdio.h so rand() scanf() would work.

Using the basic outline above I met the criteria from the lab instructions, creating my game from scratch. To ensure that I did this right I compiled my code and ran it several times looking out for wrong outputs.

# Testing

1. To make sure the results of the solution were correct, I compiled to make sure I didn’t get any errors, and then I went back and ensured that I added comment to every line to make the code work, and what each line meant and that it made sense. I then I ran the program to make sure it worked effectively again.

# Comments

1. I added the stdlib.h and the got the remainder of the value and added 1 (rand() % 4 + 1) so that way it only printed out the 4 variables inside my int array and didn’t give me any compile errors because of a wrong value, I also you the srand with the time function the randomization was the same each time the game was run making for a truly random game.
2. I kept track of how much time the user had left by adding the current and pervious time together and then checking to see if it was less than the game time, I also kept track of which buttons where being pressed to ensure the user pressed the right ones, Finally I kept track of how many rounds had been passed through so, at the end of the game I could inform the user of how far they got.
3. I only scanned in the buttons user need to use, and so every other button would not mess with my code, giving an unwanted input into the program

/\*-----------------------------------------------------------------------------

- SE 185 Lab 06

- Developed for 185-Rursch by T.Tran and K.Wang

- Name: CHimzim Ogbondah

- Section: D

- NetID: ogbondah

- Date:

-----------------------------------------------------------------------------\*/

/\*-----------------------------------------------------------------------------

- Includes

-----------------------------------------------------------------------------\*/

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <string.h>

/\*-----------------------------------------------------------------------------

- Defines

-----------------------------------------------------------------------------\*/

#define TRUE 1

#define FALSE 0

/\*-----------------------------------------------------------------------------

- Prototypes

-----------------------------------------------------------------------------\*/

int start\_menu**(**int start**);**

int buttonPress**(**int T**,** int C**,** int x**,** int S**,** int answer**);**

int time\_check**(**int currentTime**);**

int printButton**(**int values**);**

/\*-----------------------------------------------------------------------------

- Implementation

-----------------------------------------------------------------------------\*/

int main**()** **{**

int round **=** 0**;**

int TRIANGLE**,** SQUARE**,** CIRCLE**,** CROSS**;** //defining the variables that the controller will later scan in

int Triangle1**,** Circle2**,** Cross3**,** SQUARE4**;** //initilizing the variables that will equate to the corresponding values inside of the int array

int t**,** accpetedOrNot**;** //defining t so it can be scanned in by the contrller and accpetedOrNot for my time\_check function

int randmod **=** rand**()** **%** 4 **+** 1**;** //this will allow me to randomize my array without having to do the calculations over and over

int currentTime**,** prev\_Time**;** //defining currentTime and prev\_Time to be used as well in my time\_check function.

int wait **=** 0**;** //meaning less vairable just meant to loop before game starts

int buttonsToPress**[**4**]** **=** **{**1**,** 2**,** 3**,** 4**};** // Triange is 1 Circle is 2 Cross is 3 Square is 4

printf**(**"Please press the Circle button to begin the game\n"**);** //tells the user to press the circle button to begin the game

scanf**(**"%d, %d, %d, %d, %d"**,** **&**t**,** **&**TRIANGLE**,** **&**CIRCLE**,** **&**CROSS**,** **&**SQUARE**);** //time and buttons for the controller are scanned in

**while** **(**start\_menu**(**CIRCLE**)** **==** 0**)** **{** //loops until the user presses the circle button making it false

wait **=** wait **+** t**;**

**}**

**while** **(**start\_menu**(**CIRCLE**)** **==** 1**)** **{** //starts the game

int game **=** 1**;** //sets game as 1 so it can be used once this while loop breaks

**if** **(**game **==** 1 **&&** CIRCLE **==** 0**)** **{** // waits for the circle button to be unpressed before exiting the loop

**break;**

**}**

**else** **{**

**return** 1**;** //if circle button still being pressed it loops again

**}**

**}**

srand**(**time**(NULL));**

**while** **(**game **==** 1**)** **{** //The actaul game while loop

**while** **(**TRIANGLE **==** 1 **||** CIRCLE **==** 1 **||** CROSS **==** 1 **||** SQUARE **==**1**)** **{**

scanf**(**"%d, %d, %d, %d, %d"**,** **&**t**,** **&**TRIANGLE**,** **&**CIRCLE**,** **&**CROSS**,** **&**SQUARE**);**

**}**

**do** **{**

scanf**(**"%d, %d, %d, %d, %d"**,** **&**t**,** **&**TRIANGLE**,** **&**CIRCLE**,** **&**CROSS**,** **&**SQUARE**);**

**}while(** TRIANGLE **==** 0 **&&** CIRCLE **==** 0 **&&** CROSS **==** 0 **&&** SQUARE **==** 0**);**//sets the random number inside of the int array

PrintButton**(** **(**ButtonstoPress**[**randmod**])** **);** //programs prints the button to press to the user

int answer **=** PrintButton**[**randmod**];**

**if** **(**round **==** 0**)** **{** //if it is the first round then it prints out the time at zero

time\_check**(**0**);**

**}**

**if(** **(** **(**buttonPress**(**TRIANGLE**,** CIRCLE**,** CROSS**,** SQUARE**)** **&&** time\_check**(**t**))** **&&** **(**round **>** 0**)** **)** **)** **{** //another if statement allows this to iterate as well checking to see if the right button was pressed

round **=** round **+** 1**;** //if corrected 1 is added to the round

**}**

**else** **if** **(** **(**buttonPress**(**TRIANGLE**,** CIRCLE**,** CROSS**,** SQUARE**)** **==** 1**)** **&&** **(**time\_check**(**t**)** **==** 0**)** **){** //if you answer correct but the time ran out

printf**(**"Sorry you ran out of time! "**);** // it pritns that you ran out of time and how many rounds you made it to

printf**(**"You made it %d rounds\n"**,** round**);**

**break;** //exits the statement

**}**

**else** **if(** **(**buttonPress**(**TRIANGLE**,** CIRCLE**,** CROSS**,** SQUARE**)** **==** 0**)** **&&** **(**time\_check**(**t**)** **==** 1**)** **)** **{** //if you answered wrong

printf**(**"Sorry you answered incorrectly, you made it %d rounds\n"**,** round**);** //prints out that you answered incorrectly and how many rounds you made it to

**break;** //exits the statement

**}**

**else** **if** **(**round **>=** 25**)** **{** // if you make it to round 25 it announces to the user that they won the game

printf**(**"Congrats you won the game!\n"**);**

**break;** //exits the statement

**}**

**else** **{**

printf**(**"Sorry you answered incorrectly, you made it %d rounds"**,** round**);** //prints how may rounds the user made it to

**break;**

**}**

**return** 0**;** //returns zero if the user wins the game or lose which exits the game loop

**}**

**}**

int start\_menu**(**int start**)** **{** //start menu function

**if(**start **==** 1**)** **{** // returns true if the circle button on the controller was pressed

**return** TRUE**;**

**}**

**else** **{**

**return** FALSE**;** //if circle was not pressed it returns false

**}**

**}**

int time\_check**(**int currentTime**)** **{** //time\_check function

int scoreTime **=** 2200**;** //sets the game time equal to 2200 miliseconds

int prev\_Time **=** 0**;** //sets the previous time to zero

int accpetedOrNot**;** //variable that will print how much time is left

**if((**currentTime **+** prev\_Time **)** **<** scoreTime **&&** **(**currentTime **+** prev\_Time**)** **>** 0**)** **{** //if the current time and the previous time are less than the score time it returns true

accpetedOrNot **=** scoreTime **-** **(**currentTime **+** prev\_Time**);** //variable is equal to how much time is remaining based off of how long the user took to answer

printf**(**" You have %d miliseconds left to answer!"**,** accpetedOrNot**);** //prints how much time is left for the user to answer

**return** TRUE**;**

**}**

**else** **if(**currentTime **+** prev\_Time**)** **>=** scoreTime**)** **{** //returns false if greater than the value

**return** FALSE**;**

**}**

**else** **if(**currentTime **==** 0**){**

printf**(**"You have %d miliseconds left to answer!"**,** scoreTime**);**

**}**

**}**

int buttonPress**(**int T**,** int C**,** int x**,** int S**,** int answer**)** **{**

**if(**T **==** 1**)** **{**

int Triangle1 **=** 1**;** //if Triangle is pressed and is equal to the random button returns true

**if** **(**buttonToPress**[**answer**]** **==** Triangle1**)** **{**

**return** TRUE**;**

**}**

**else** **{**

**return** FALSE**;**

**}**

**}**

**else** **if(**C **==** 1**)** **{**

int Circle2 **=** 2**;**

**if** **(**buttonToPress**[**answer**]** **==** Circle2**)** **{** //if circle is pressed and is equal to the random button returns true

**return** TRUE**;**

**}**

**else** **{**

**return** FALSE**;**

**}**

**}**

**else** **if** **(**x **==** 1**)** **{**

int Cross3 **=** 3**;**

**if** **(**buttonToPress**[**answer**]** **==** CROSS**)** **{** //if cross is pressed and is equal to the random button returns true

**return** TRUE**;**

**}**

**else** **{**

**return** FALSE**;**

**}**

**}**

**else** **if(**S **==** 1**)** **{** //if square is pressed and is equal to the random button returns true

int SQUARE4 **=** 4**;**

**if** **(**buttonToPress**[**answer**]** **==** SQUARE4**)** **{**

**return** TRUE**;**

**}**

**else** **{**

**return** FALSE**;**

**}**

**}**

**}**

int printButton**(** int values**)** **{**

int Triangle **=** 1**;** // set triangle to 1 to line up with the int array

int Circle2 **=** 2**;** // set circle to 2 to line up with the int array

int Cross3 **=** 3**;** // set cross to 3 to line up with the int array

int SQUARE4 **=** 4**;** // set square to 4 to line up with the int array

**if** **(**values **==** Triangle1**)** **{**

printf**(**"Press the Triangle button!\n"**);** //if the random button is equal to the value of triangle1 it prints press triagnle

**}**

**else** **if** **(**values **==** Circle2**)** **{**

printf**(**"Press the Circle button!\n"**);** //if the random button is equal to the value of Circle2 it prints press circle

**}**

**else** **if** **(**values **==** Cross3**)** **{**

printf**(**"Press the X button!\n"**);** //if the random button is equal to the value of Cross3 it prints press cross

**}**

**else** **if(**values **==** SQUARE4**)** **{**

printf**(**"Press the Square button!\n"**);** //if the random button is equal to the value of Square4 it prints press square

**}**

**}**