Project 1

Title

Yahtzee! V.8

Course

CIS-5

(CSC-5 46688)

Due Date

July 31, 2022

Author

Jessriel Menguito

**Table of Contents**

1. Introduction - pg. 3
2. Game Play and Rules - pg. 3
3. How to Play - pg. 3
4. Scoring - pg. 4
5. Bonus - pg. 6
6. End of the Game - pg. 6
7. My Approach to the Game - pg. 7

7.1. Translating Game Play Rules to Programming Language - pg. 7

7.2. My Similarities to the Board Game - pg. 7

7.3 The Logic - pg. 7

1. Proof of a Working Project - pg. 20
2. References - pg. 33
3. Program - pg. 33
4. Constructs & Concepts Utilized - pg. 53

**1 Introduction**

Yahtzee is a dice game made by Milton Bradley. The goal of the game is to score points by rolling five dice. The dice can be rolled up to three times in a turn. By selecting different combinations the players can set aside dice to score different combinations of points. The game has thirteen rounds and at the end of each round the player chooses which scoring category they want to use for that round. Once a category has been used, it cannot be used again. A Yahtzee is a five-of-a-kind that scores 50 points, the highest of all categories. The winner is the player who scores the most points.

**2 Game Play and Rules**

Object of the Game:

Roll dice to get the highest score after all 13 rounds.

Game Play

*In summary:*

1. *Roll up to 3 times each turn to rack up the best possible score.*
2. *Decide which dice combo you're going for.*
3. *After each turn, write your score in one empty box on the scorecard.*
4. *Each player in turn places all 5 dice in the cup, shakes the cup and rolls out the dice. Each turn consists of a maximum of three rolls. The first roll must be made with all five dice. If the player chooses to roll a second and, if desired, a third time, they may pick up any or all the dice and roll again.*

**3. How to play:**

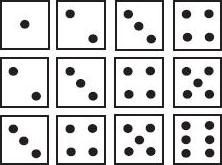
1. Roll all 5 dice.
   1. If you get the exact combo you want, mark your scorecard right away. Otherwise, continue with your 2nd roll.
2. Your second roll:
   1. Set aside any keepers' and re-roll the rest.
      1. Hate them all? Re-roll them all. If you get the exact combo you need, mark your scorecard. Otherwise, continue with your 3rd roll.
3. Your third and final roll:
   1. If you still haven't nailed a combo, or just want to maximize your score:
   2. Roll some or all of your dice (as above), even any keepers you might have set aside.
   3. You must enter a score at the end of this roll, whether or not you like your dice.
   4. If your roll doesn't suit any of your empty boxes, you have to enter a zero somewhere. BUT scoring even a single die value is better than a zero.

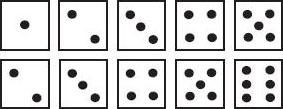
**4.Scoring**

1. There are 13 scoring boxes. On each completed turn, the player must score in one of the 13 boxes. The boxes may be filled in any order, according to the player's best judgment.
2. The score card has an upper section and a lower section:
   1. Upper Section
      1. The upper section contains six boxes labeled 1 to 6 ( "Aces", "Twos", "Threes", "Fours", "Fives" and "Sixes").
      2. When a player chooses to score in the Upper Section,theycounts and add only the dice with the same number and enter the total of these dice in the appropriate box.
      3. If a player, on his turn, rolls and elects to take his score in the Upper Section,they would enter 9 in the "Threes" box.
      4. But they are also allowed to score a 2 in the "Twos" Box or a 4 in the "Fours" box. The players can also score a zero in th
      5. e Aces box.
   2. Lower Section
      1. In the lower section, you score for various combinations and they are played exactly as indicated:
      2. 3 of a kind
         1. The "3 of a kind" box may be filled in only if the dice show at least 3 of the same number. For example:
            1. This would score 18 (total of all dice).
      3. 4 of a kind
         1. Score the total of all dice provided they include 4 dice of the same number. For example:
            1. This would score 14 (total of all dice) in the "4 of a kind" box. (You may also score in "3 of a kind" or in the upper section).
      4. Full House
         1. You need to roll both a "3 of a kind" and a pair. A "Full House" scores 25 points. (You may also score in "3 of a kind" or in the upper section).



* + 1. Small Straight
       1. This is any sequence of four numbers, such as:



* + 1. The 5th die can be any number. Any "Small Straight" is valued at 30 points.
  1. Large Straight
     1. This is a sequence of five numbers, either:
        1. A "Large Straight" is valued at 40 points.
        2. (You may also score a Large Straight in the Small Straight box).
  2. Chance
     1. This offers a player the opportunity to score on any turn where they do not choose to score in any of the other open boxes. Mark scores by totalling the number of points on all dice.
     2. This combination scores 24 points.



Yahtzee

This is any "five of a kind". Score 50 points in the "Yahtzee" box.



**5.Bonus**

1. You get a bonus of 35 points if the total number of points you scored in the upper section is 63 or higher. (For quick calculation, 63 may be reached by scoring 3 "Aces", "Twos", and so on through "Sixes").
   1. A bonus can be obtained by having a total of 63 points or more scored in any manner in the Upper Section
2. If you roll multiple Yahtzee, you can rack up a seriously high score.
   1. If you roll a Yahtzee and you have already filled in the Yahtzee box with a 50 , you get a 100-point bonus!
   2. Take one bonus chip, then place a checkmark in the Yahtzee bonus box . Then fill in one of the 13 boxes on your score card according to the Joker Rules (see below).

**6.End of the Game**

The player with the highest grand total at the end of all 13 rounds is the winner.

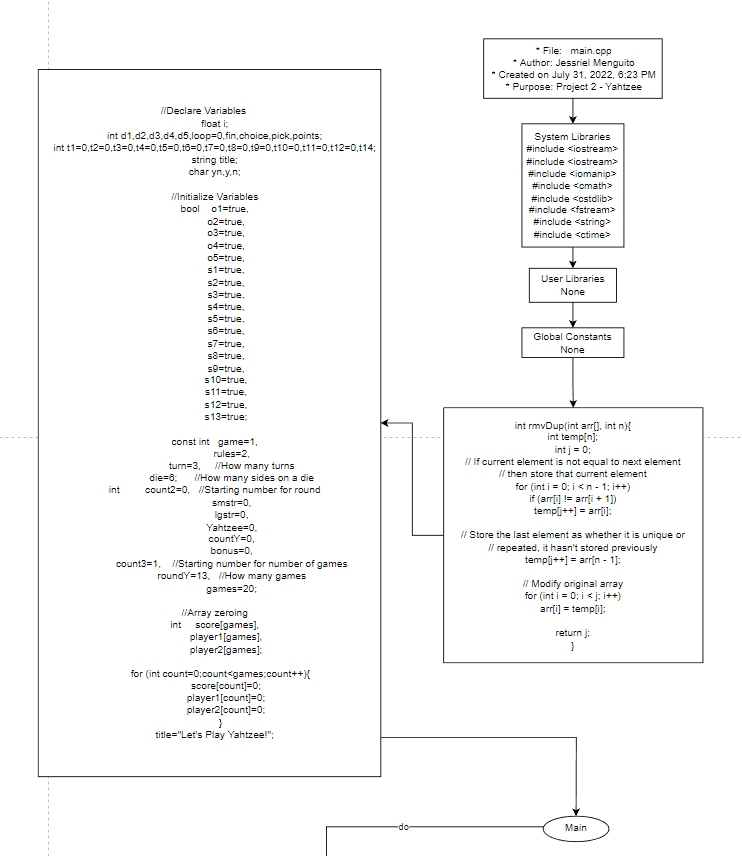
**7.My Approach to the Game**

**7.1 Translating Game Play Rules to Programming Language**

My main concerns when developing the program for this game was the scoring. The majority of the work will be how to interpret the random input of 6 integers. At first, my solution was to use functions and bools to determine if the integers matched to a score and if it were true, output the value for the user to choose. Upon further revisions, I realized how difficult it was to sort and organize the information from the six integers when determining the correct output so I shifted to arrays. With arrays I was able to sort the integers in order and create simple if statements to check values.

**7.2 My Similarities to the Board Game**

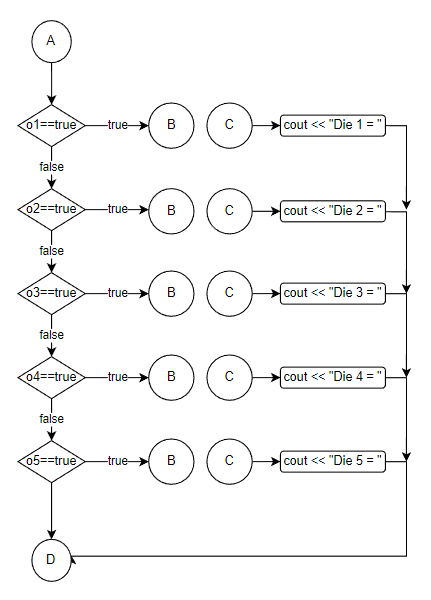
My goal was to automate the rolling and scoring of the game. After all the dice rolling decisions have been made. The player is shown what choices are available to them. After selecting which score works best for them, the program automatically rolls the next set of dice and repeats for all 13 turns. As the player selects each score, the total is continuously added between rounds and is displayed at the end.

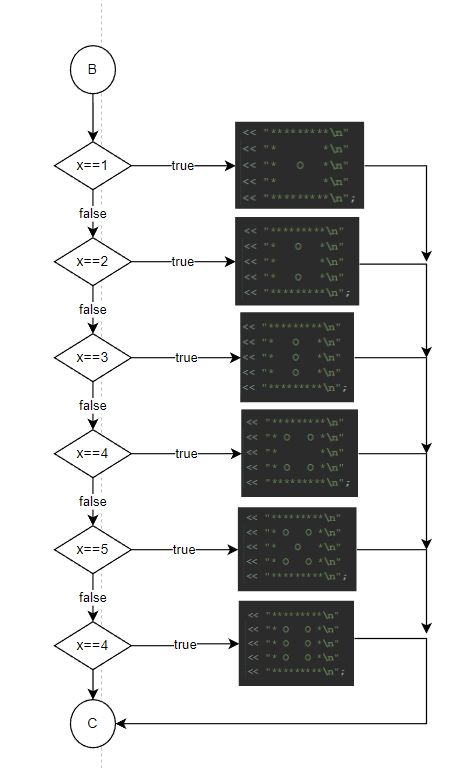
At this current iteration, the program works for a single player. It basically will take the place of a player's dice roll and score keeper. 

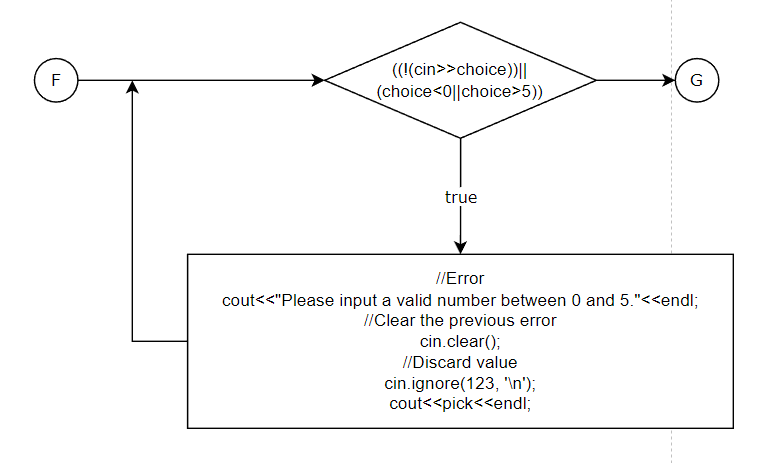
**7.3 The Logic**

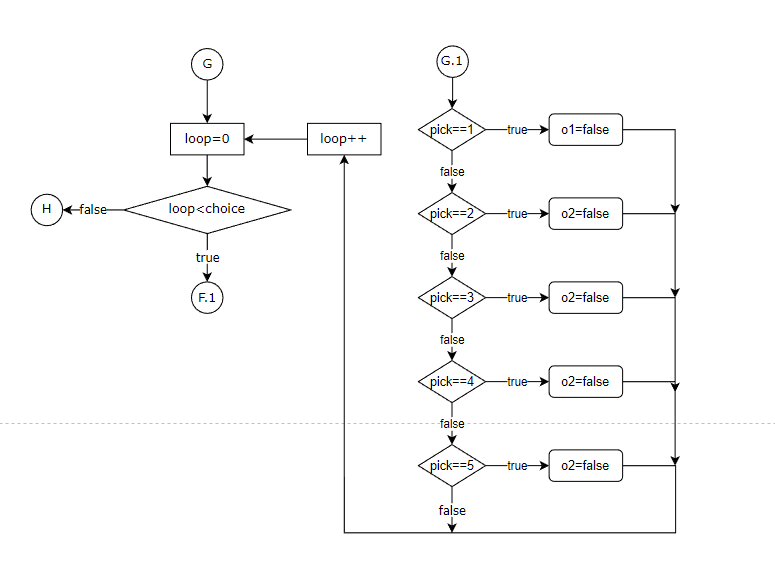
Flow Chart

For readability, I have broken up my flow chart in several sections.

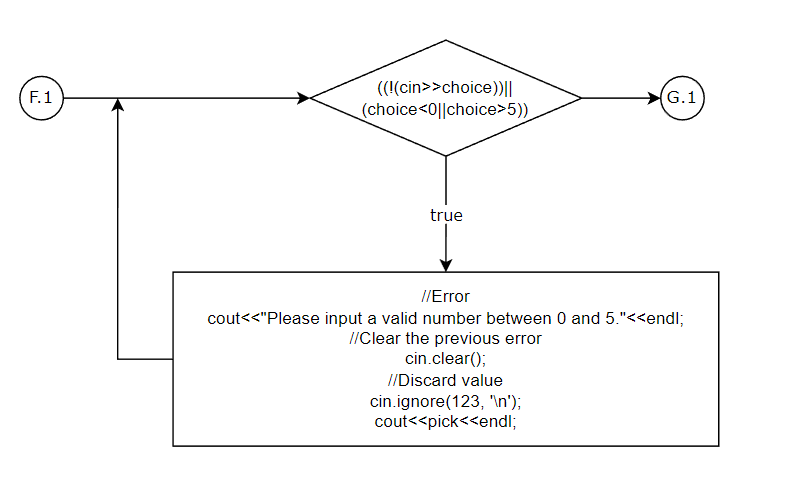
* Opening Comments
* Bring in system libraries
* Enter main
* Declare all variables. Initiate required variables.
* Roll each die with bool true
* Call function where if true show die and ASCII art
* Return to the for loop at “D”.

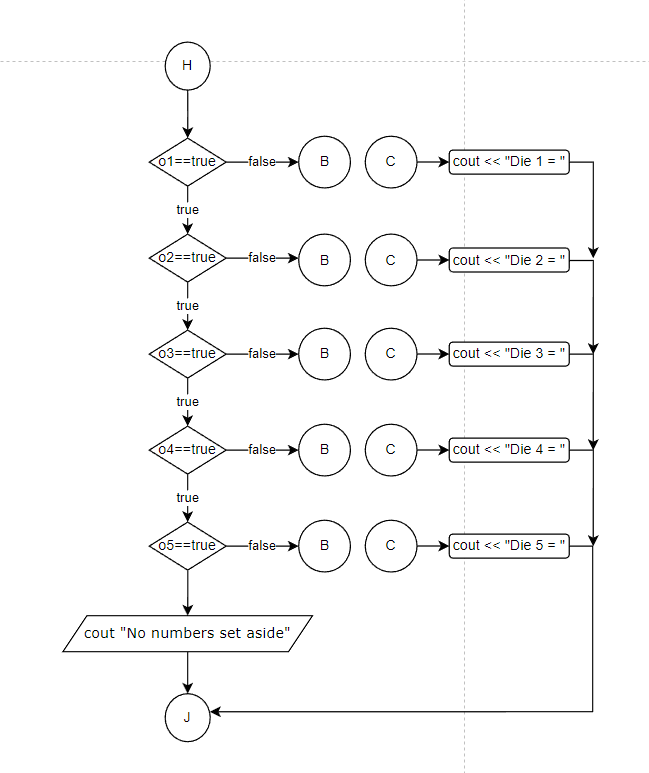


* While for loop is true:
* 1. Get input from the player how many dice they want to set aside and verify input.

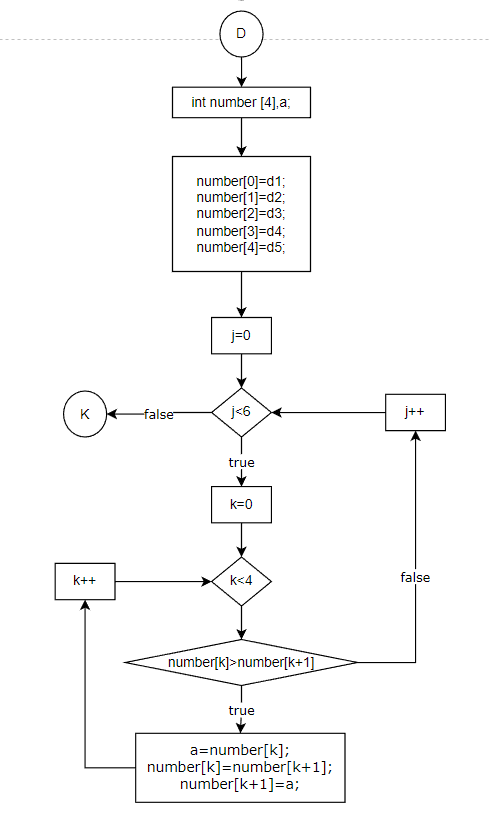


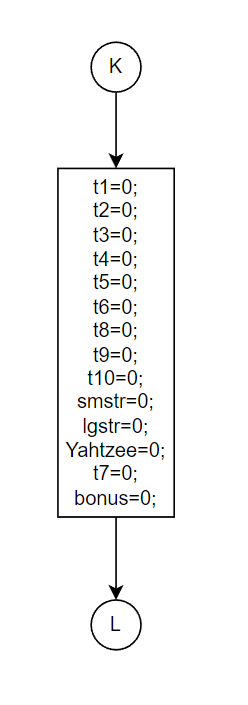
* Start for loop where it loops equal to the previous input. Then ask the player which die they want to set aside, verifying each time.



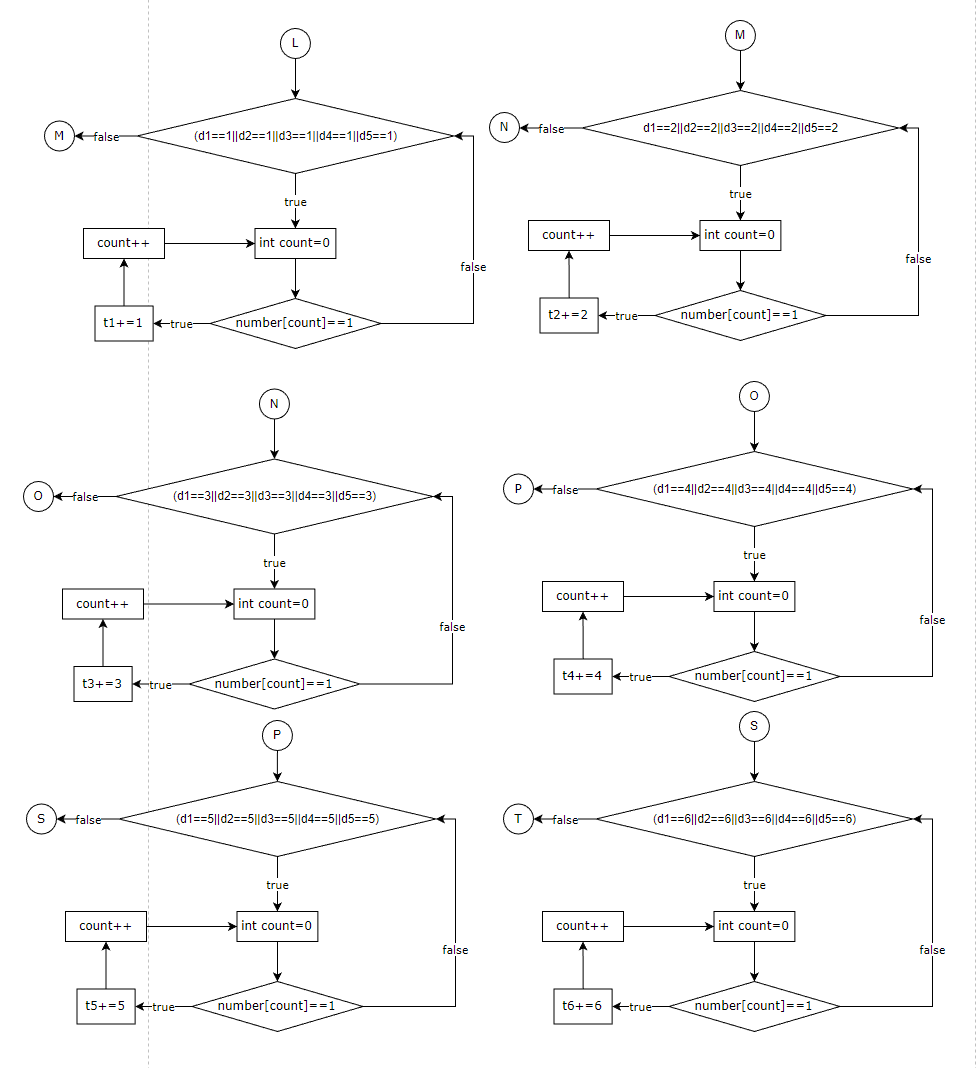


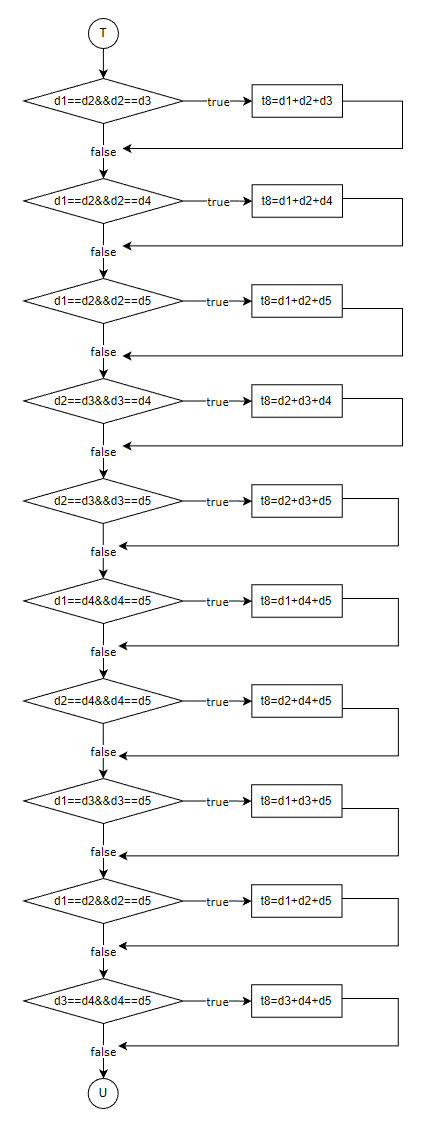
* At “J” return to the for loop and contine loop until “count2+1<turn”. Then “count2++”.
* Show the player which dice(die) they have chosen to set aside via function where bools equal false.



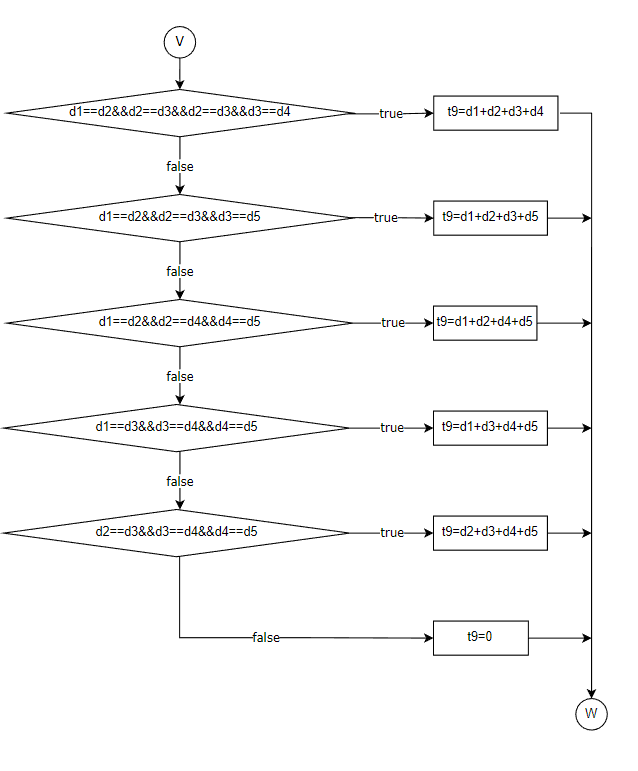


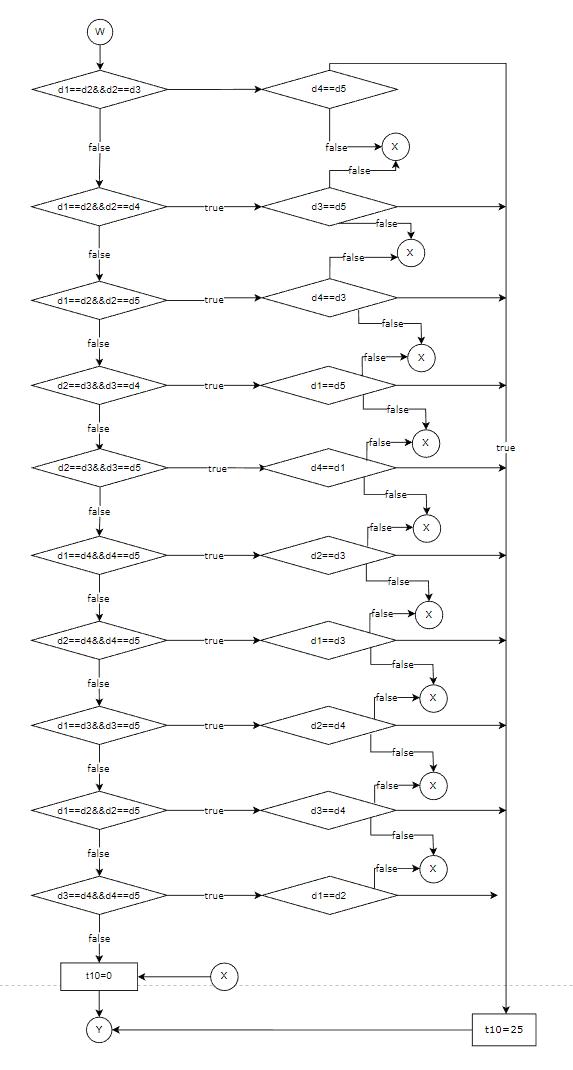
* Transfer all dice numbers to “number” array.
* Then sort the numbers from smallest to largest. This will be necessary when we need to determine small straights and long straights in the scoring.
* Reset all variables to 0 if this is not the first time through the loop.
* Using a for loop to check each value in the “number” array and if it equals to 1, add it to “t1”. This is repeated for numbers 2-6.

****

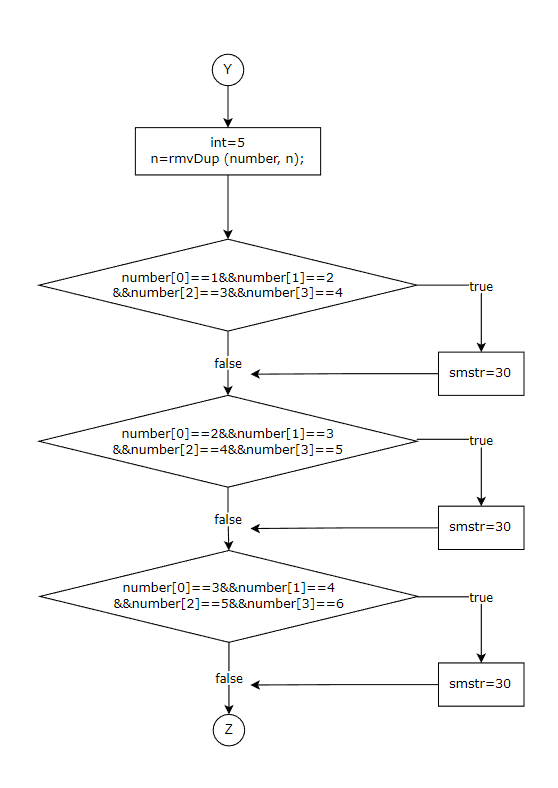


* Using brute force, I checked each possible combination of every triplet. If the triplet was found, I then added all values together.
* Using brute force, I checked each possible combination of every four-of-a-kind.. If the found, I then added all values together.

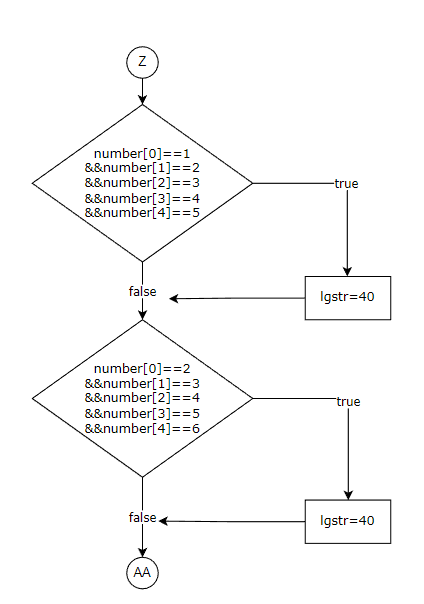




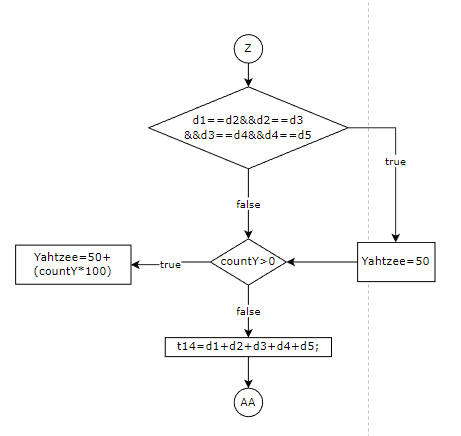
* Using the previous method to check triplets, I added an if statement to check if the remaining numbers are pairs. If true then, “t10” will equal 25.



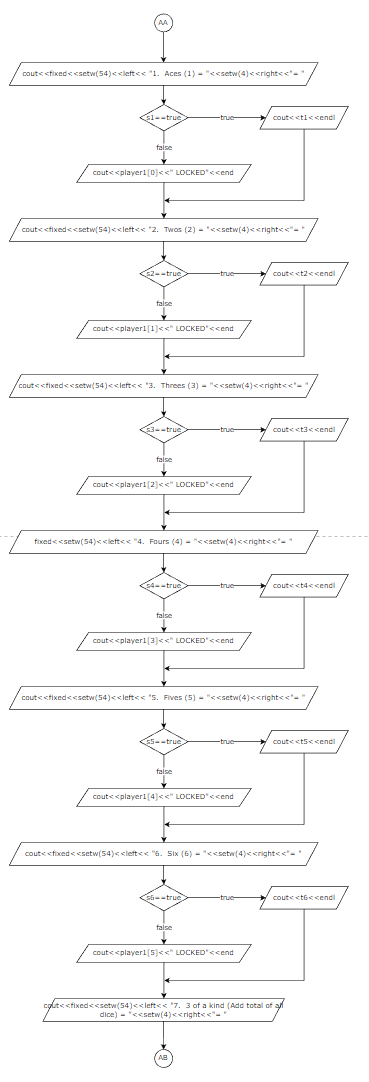
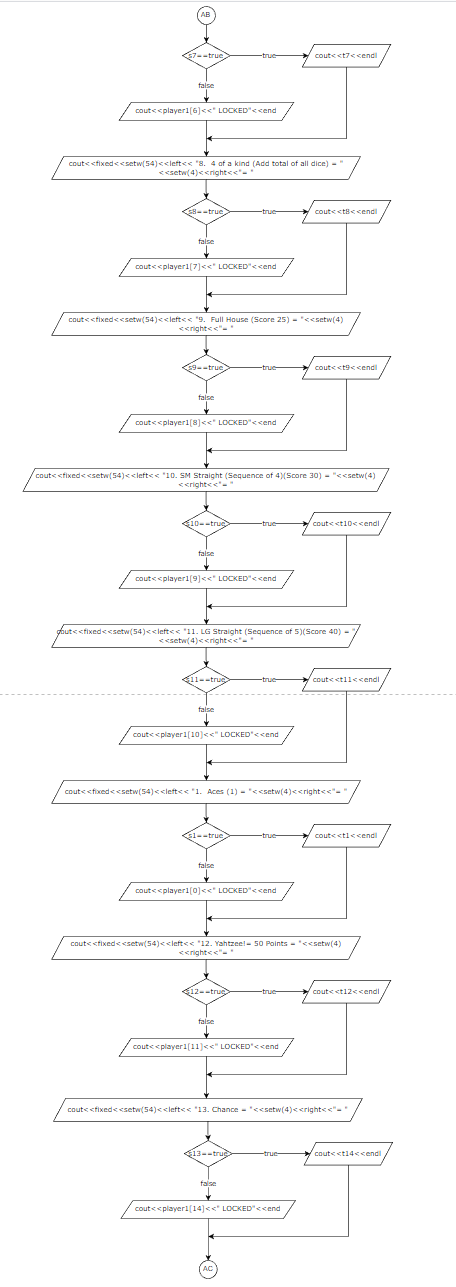
* Using the number array. I checked if the values, which were already ordered numerically, match a continuous “small straight”.



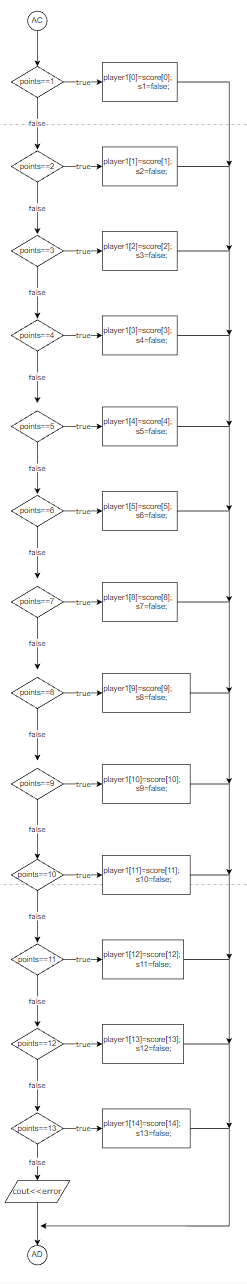
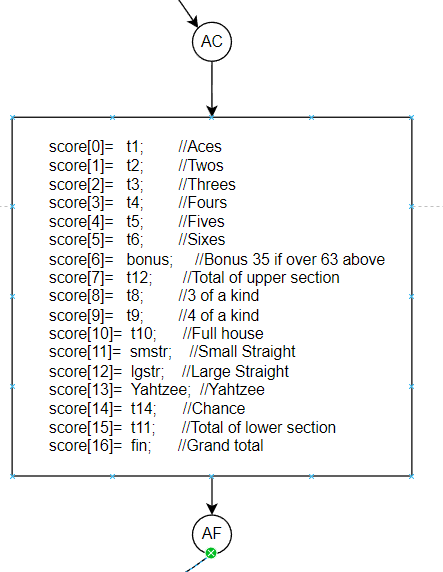
* Like above, I checked if the values match a continuous “large straight”.



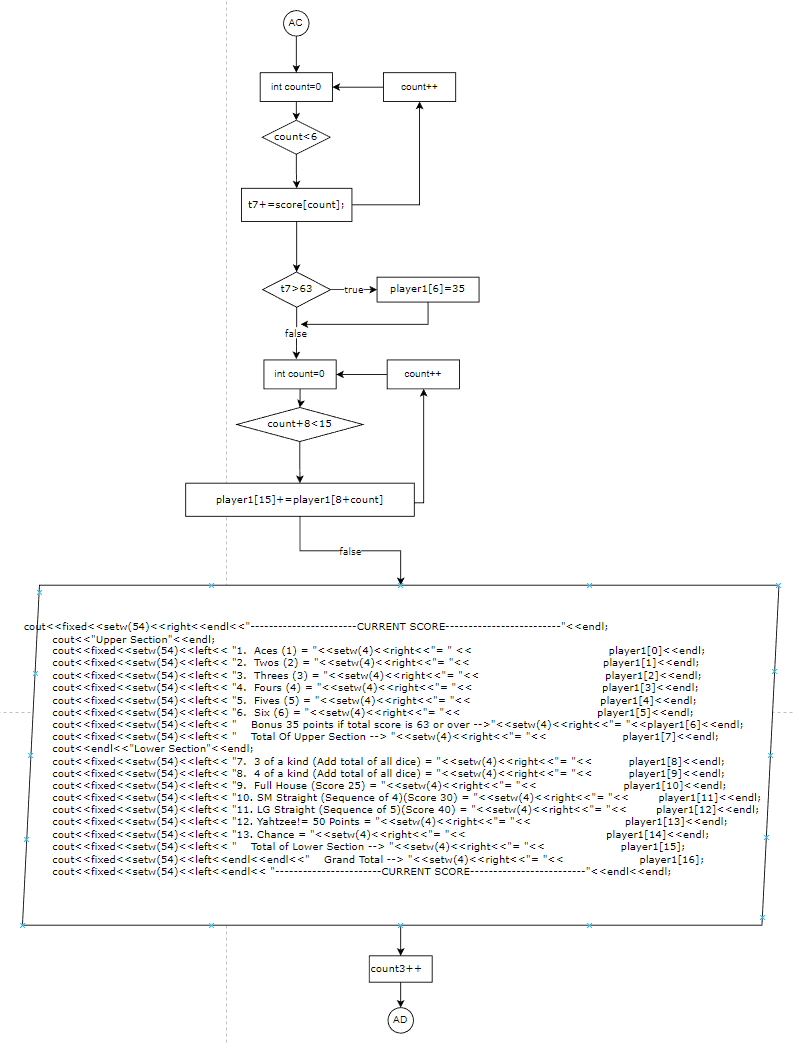
* The program checks each value against each other. If all values match, then a Yahtzee is scored. If more than one Yahtzee is score, an additional 100 points is earned.
* After Yahtzee, the Chance value is added to “t14”.



* After calculating each possible score, the program then shows the player which scores they have not “LOCKED” yet. When the code loops, one or more of these values will display “LOCKED” signifying that value is saved.

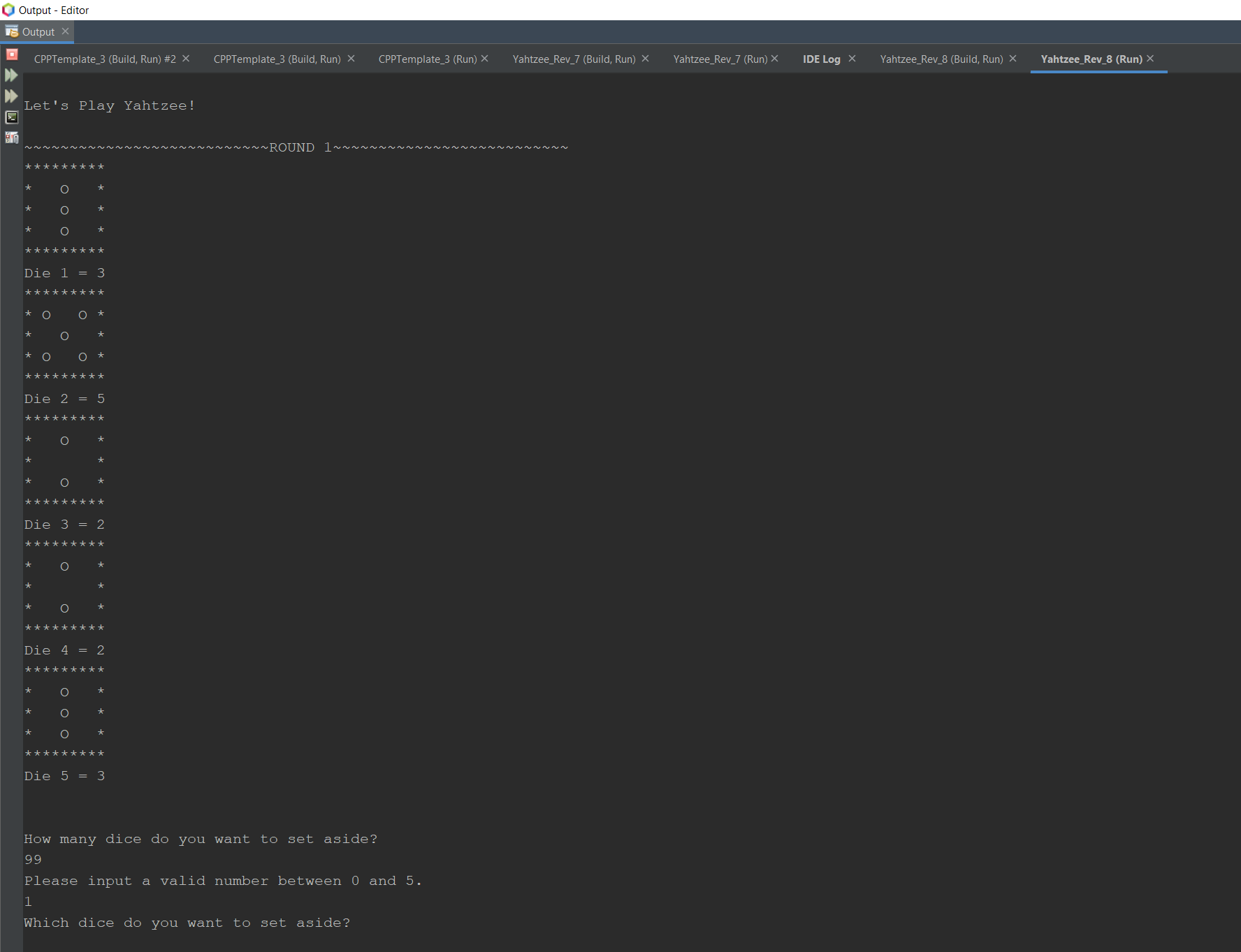


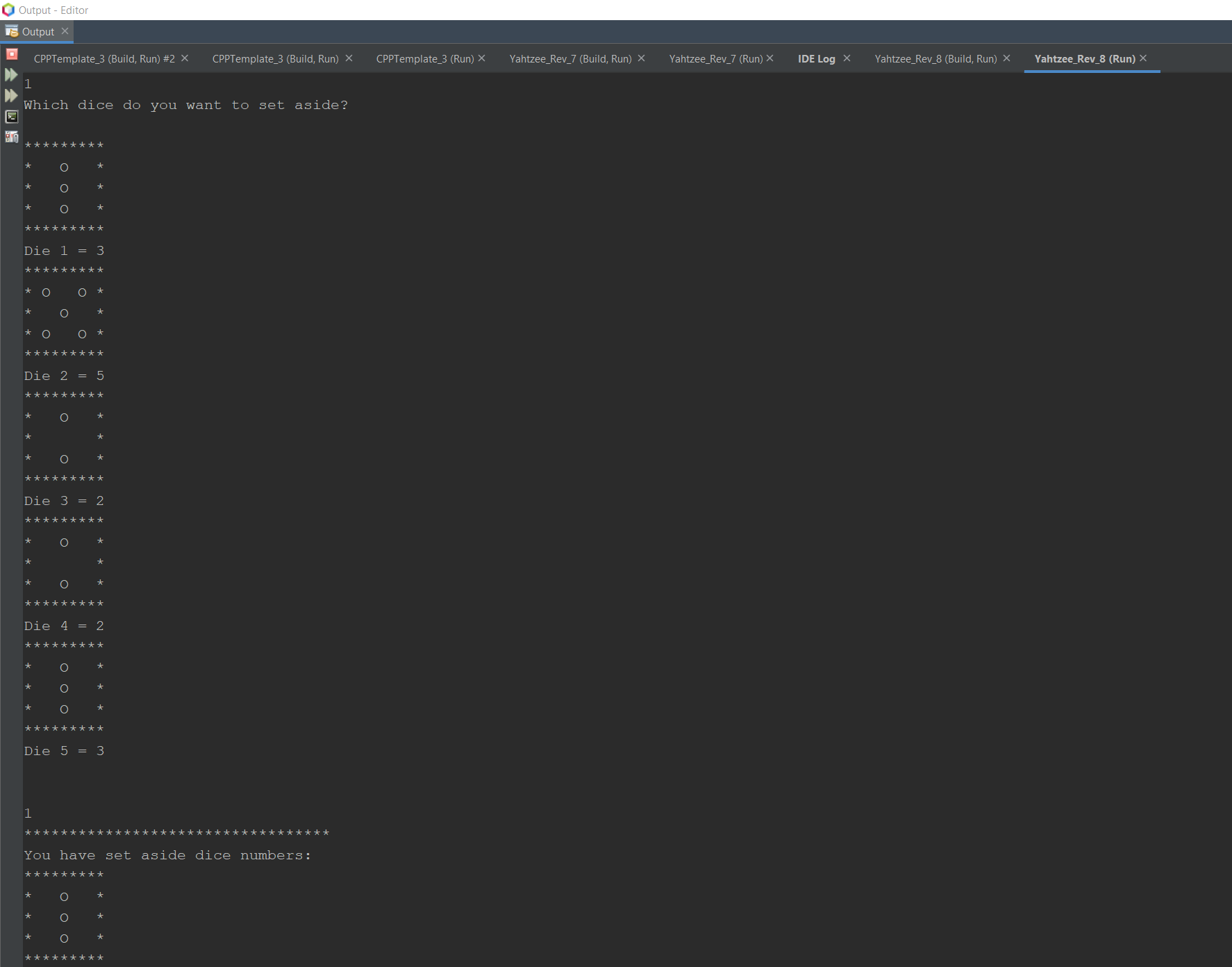
* All scores earned this round are then applied to a temporary array.
* The program then takes an input from the player. Based on that input, the relevant score is then set to false and that value is then transferred from the temporary array to that player’s array.
* The final scoring is added where the program checks to see if the upper group of values are greater than 63. If true, then the player gains an extra 35 points. The lower scores are added together and the program outputs all scores saved in that player’s array.

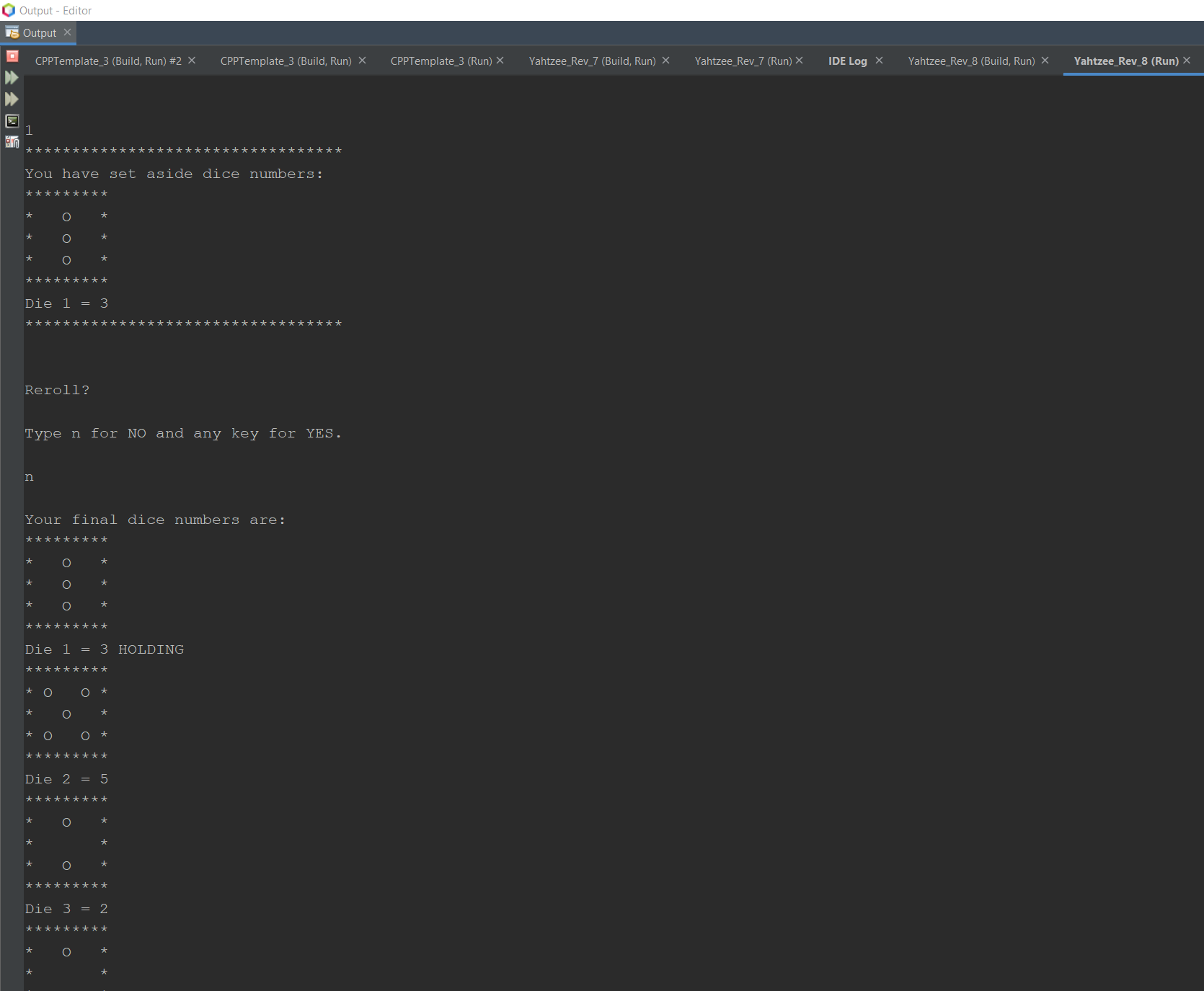


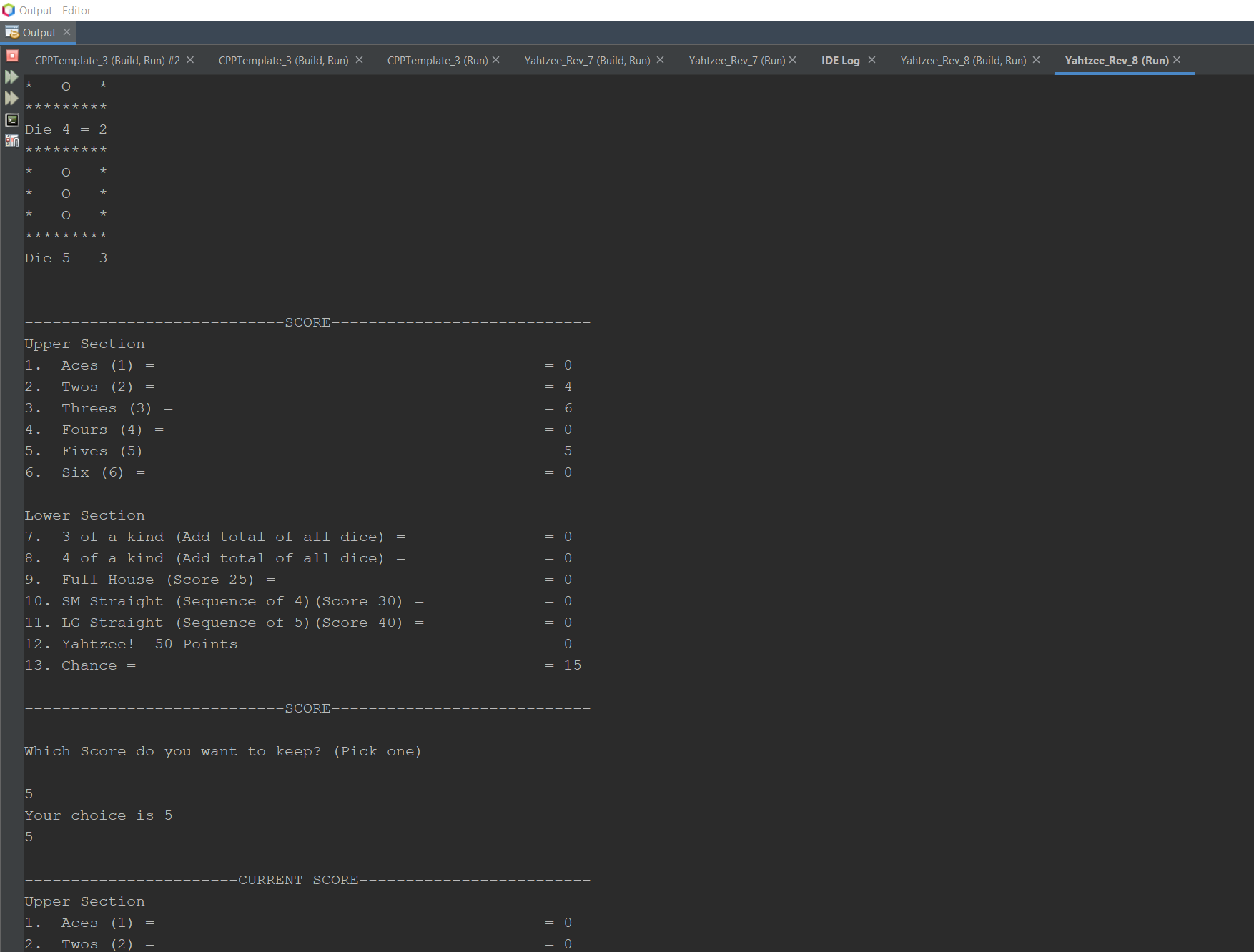
* The program then loops back at “AC” and repeats until all 13 turns are completed.

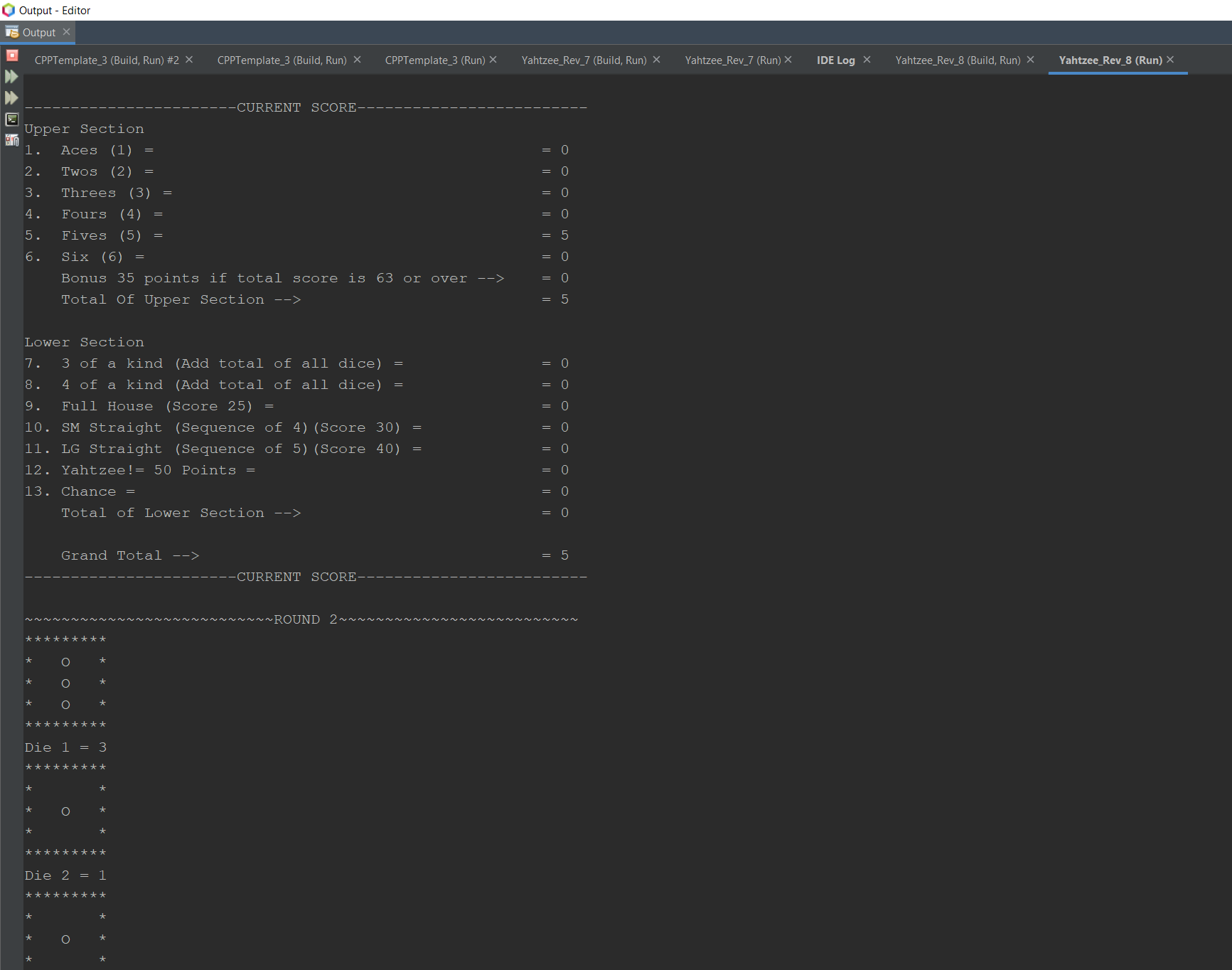
**8. Proof of a Working Project**

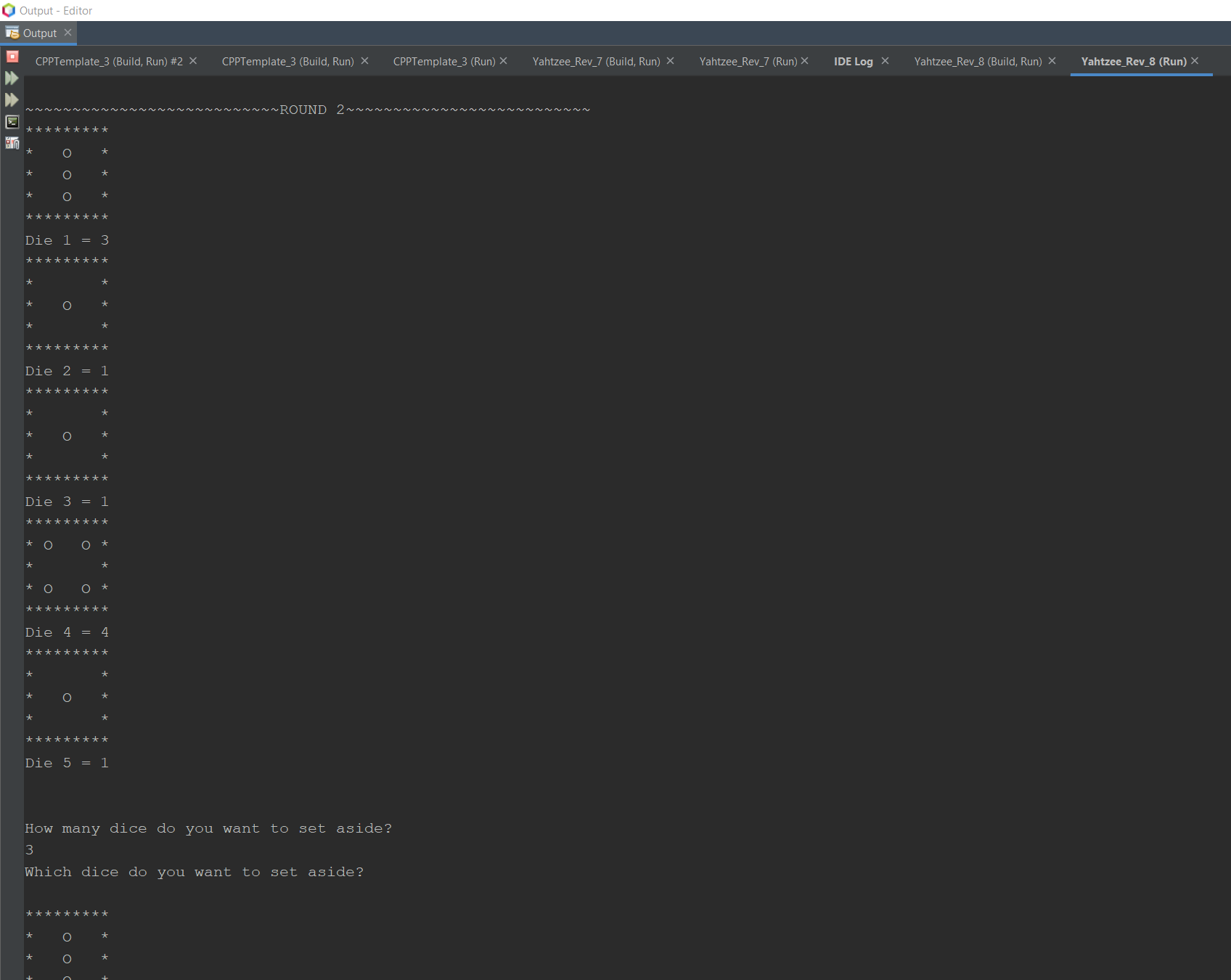


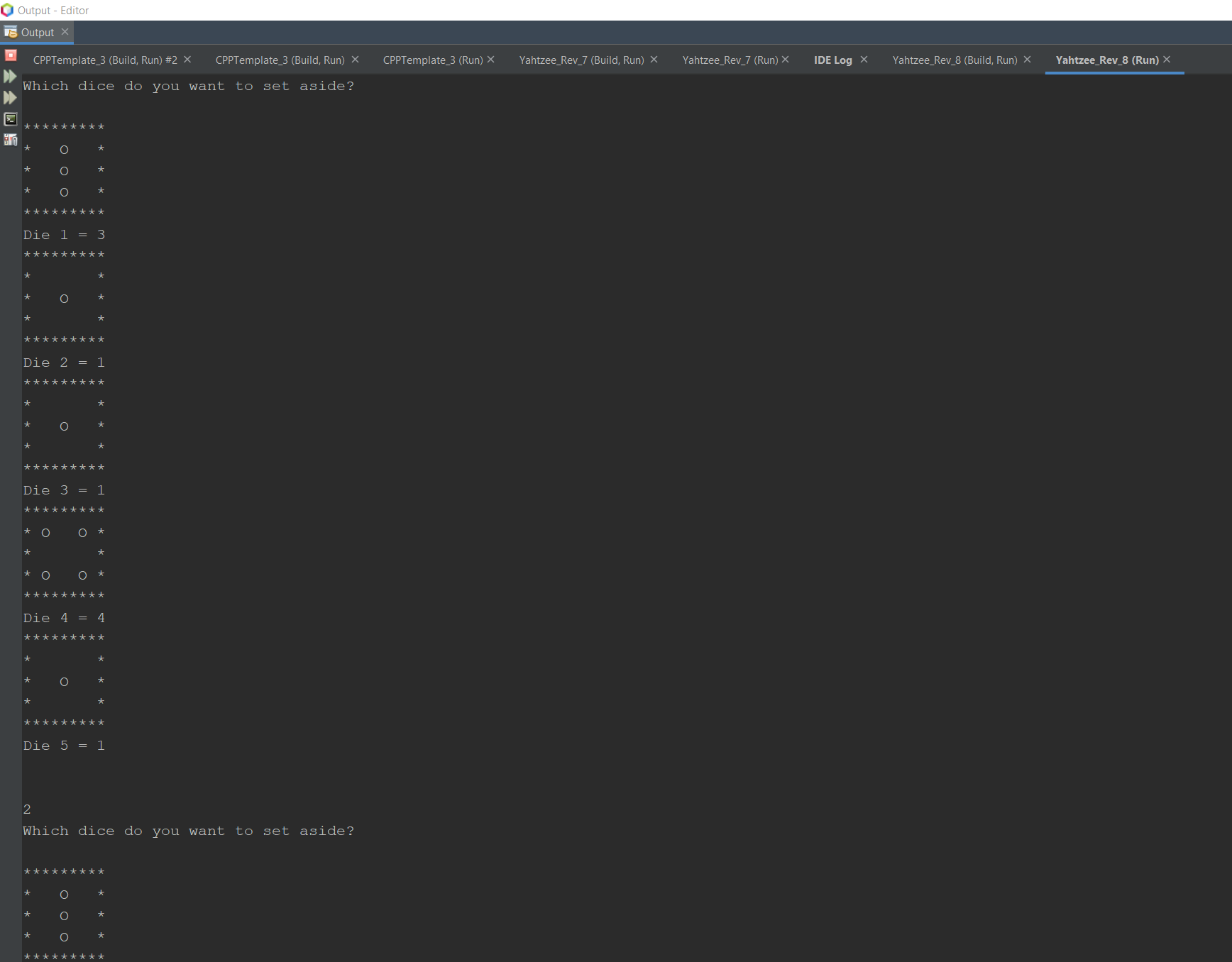


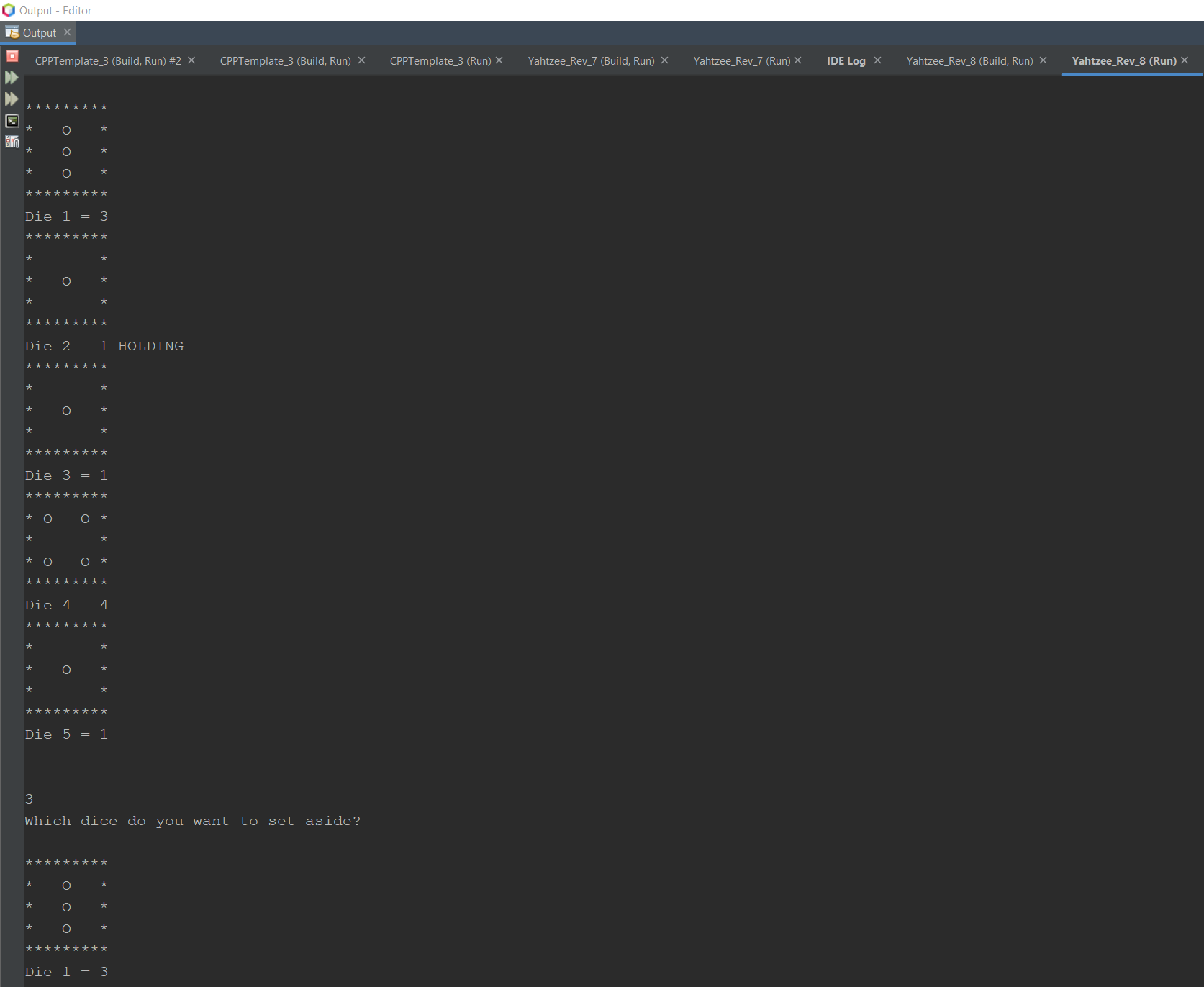


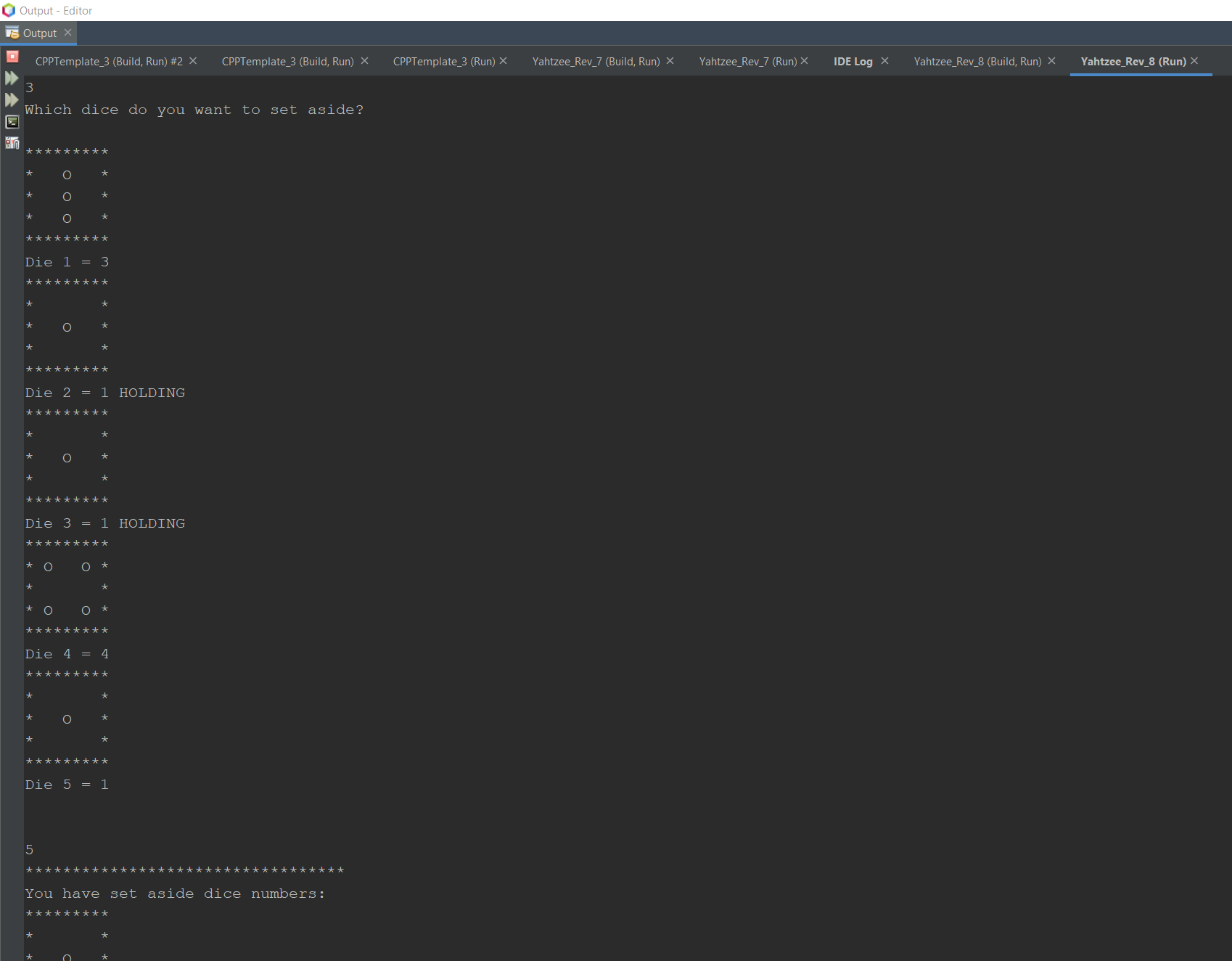


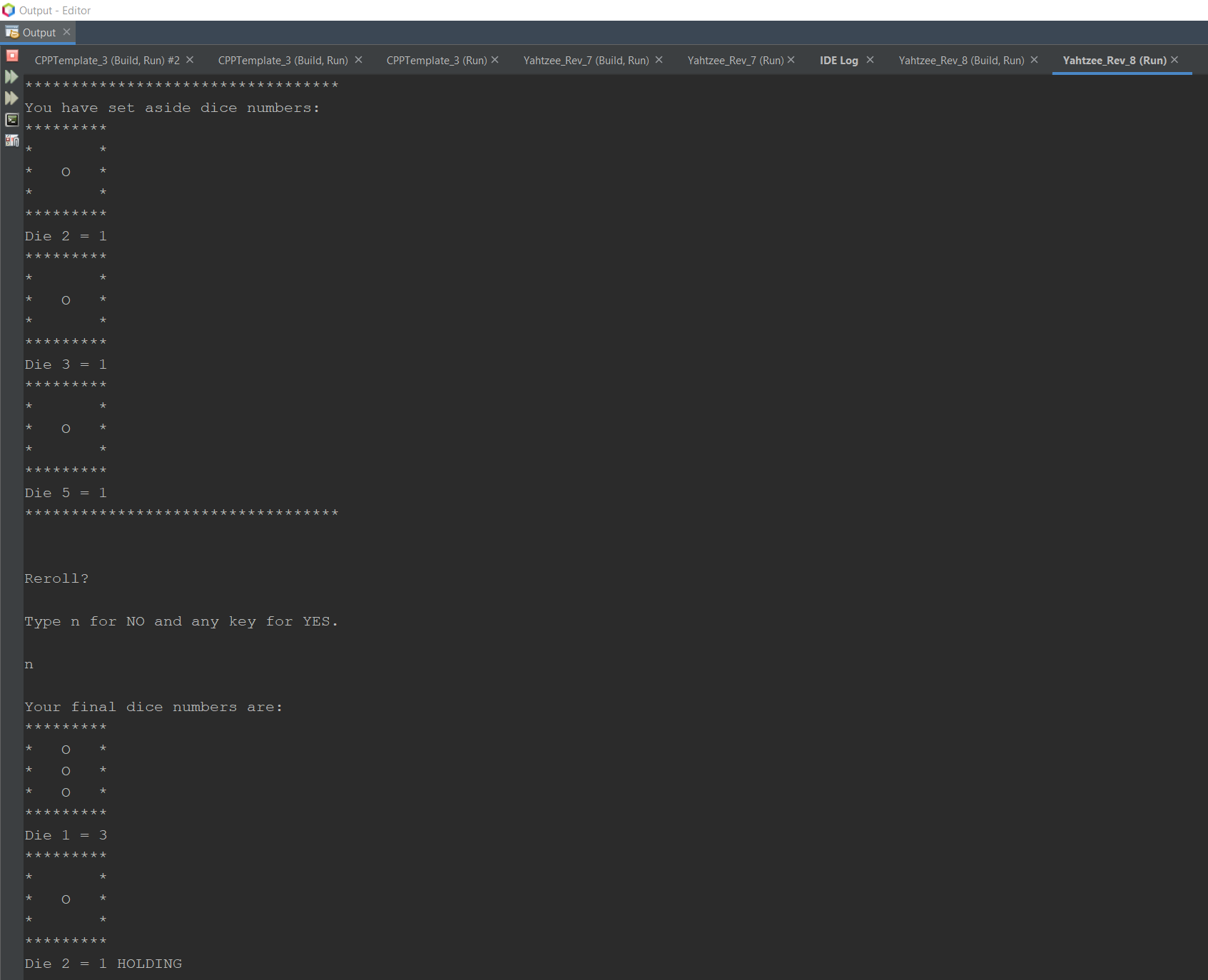


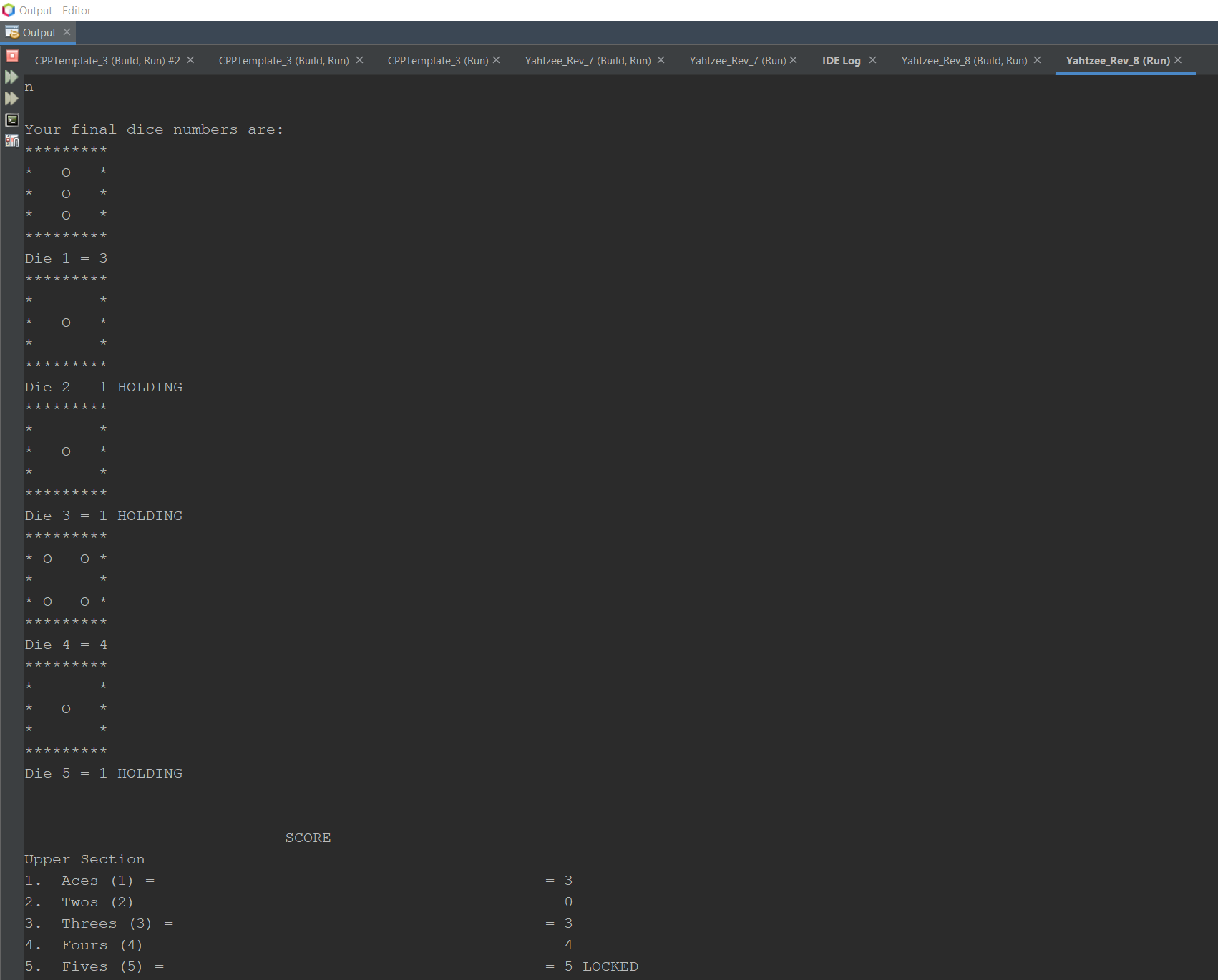


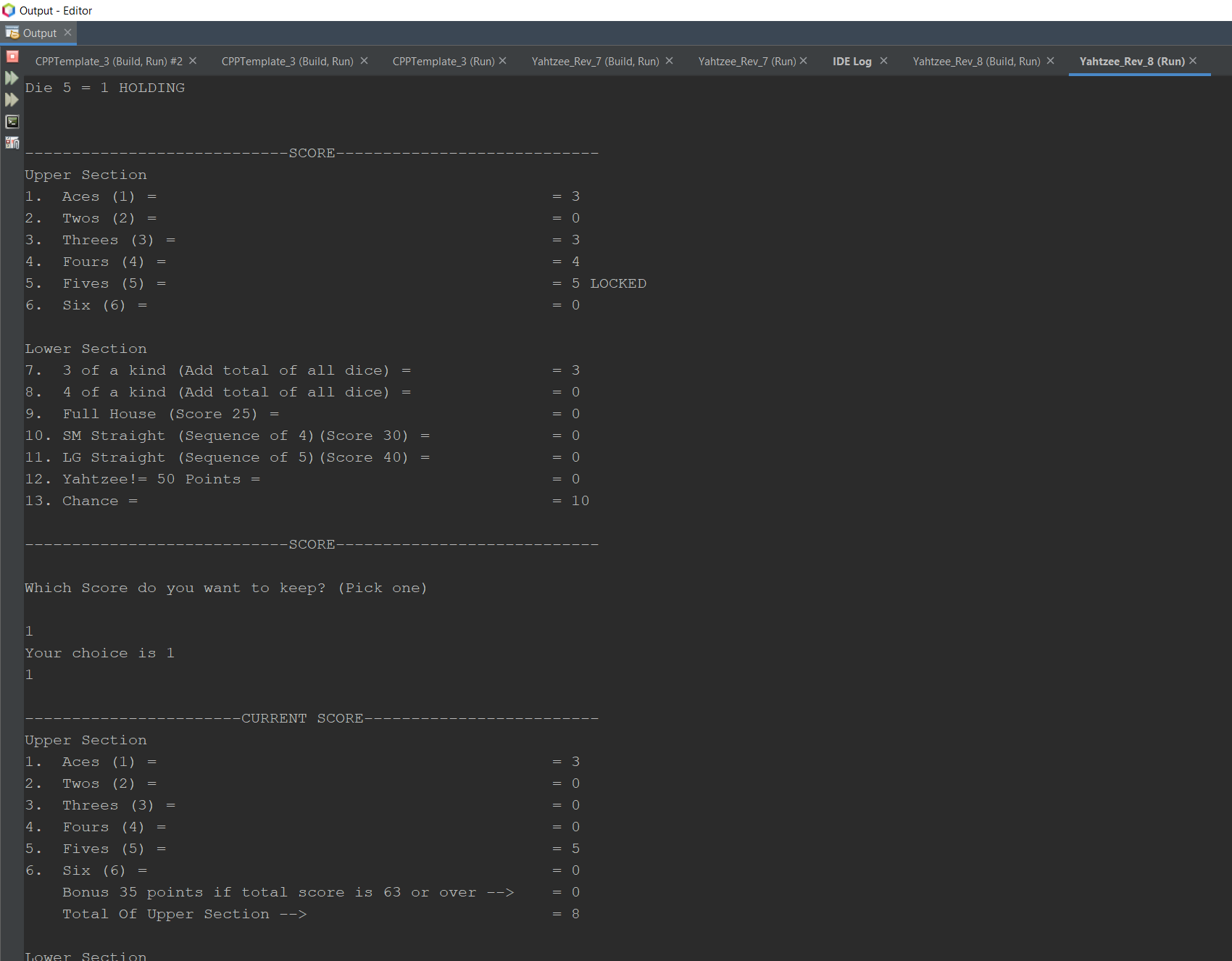


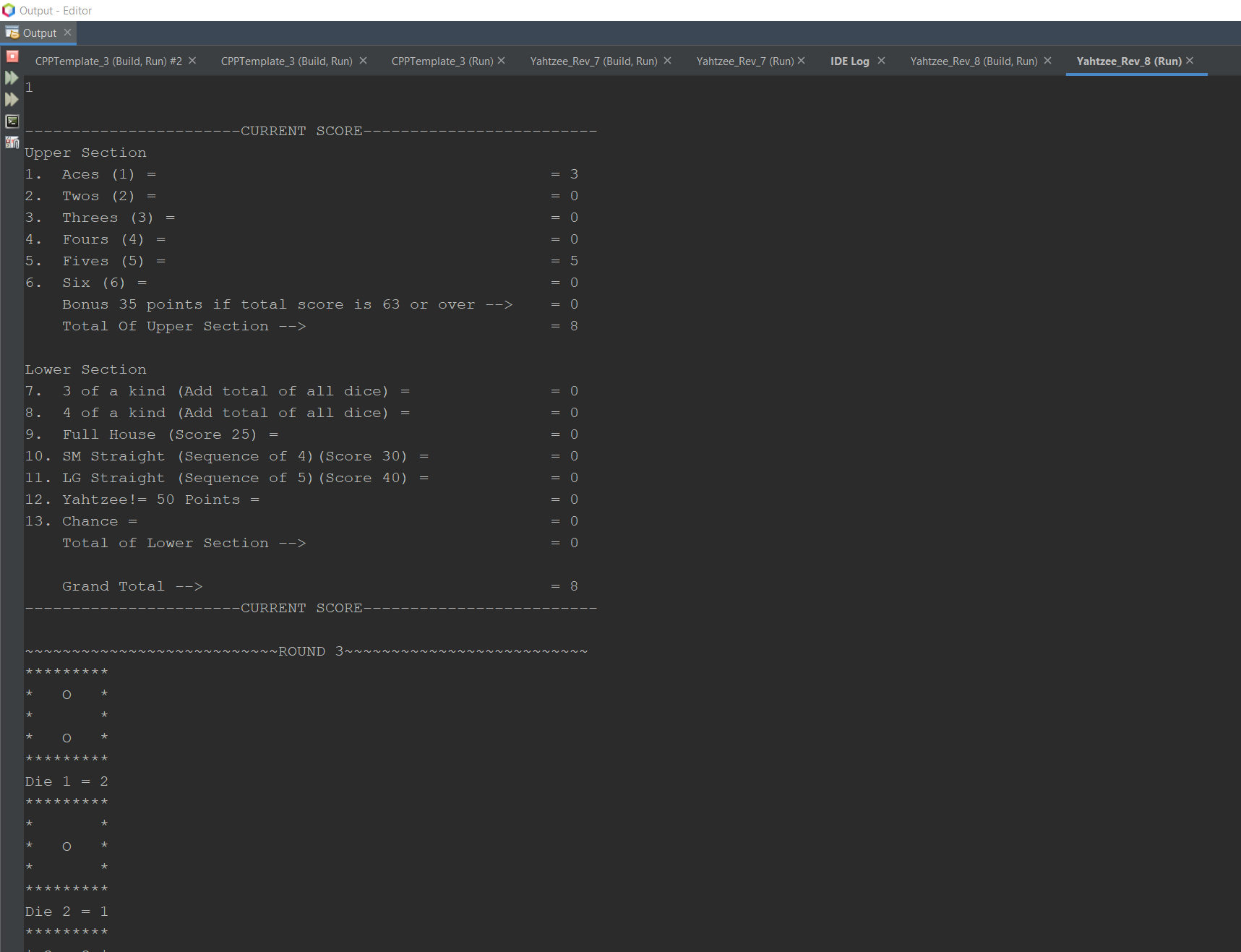




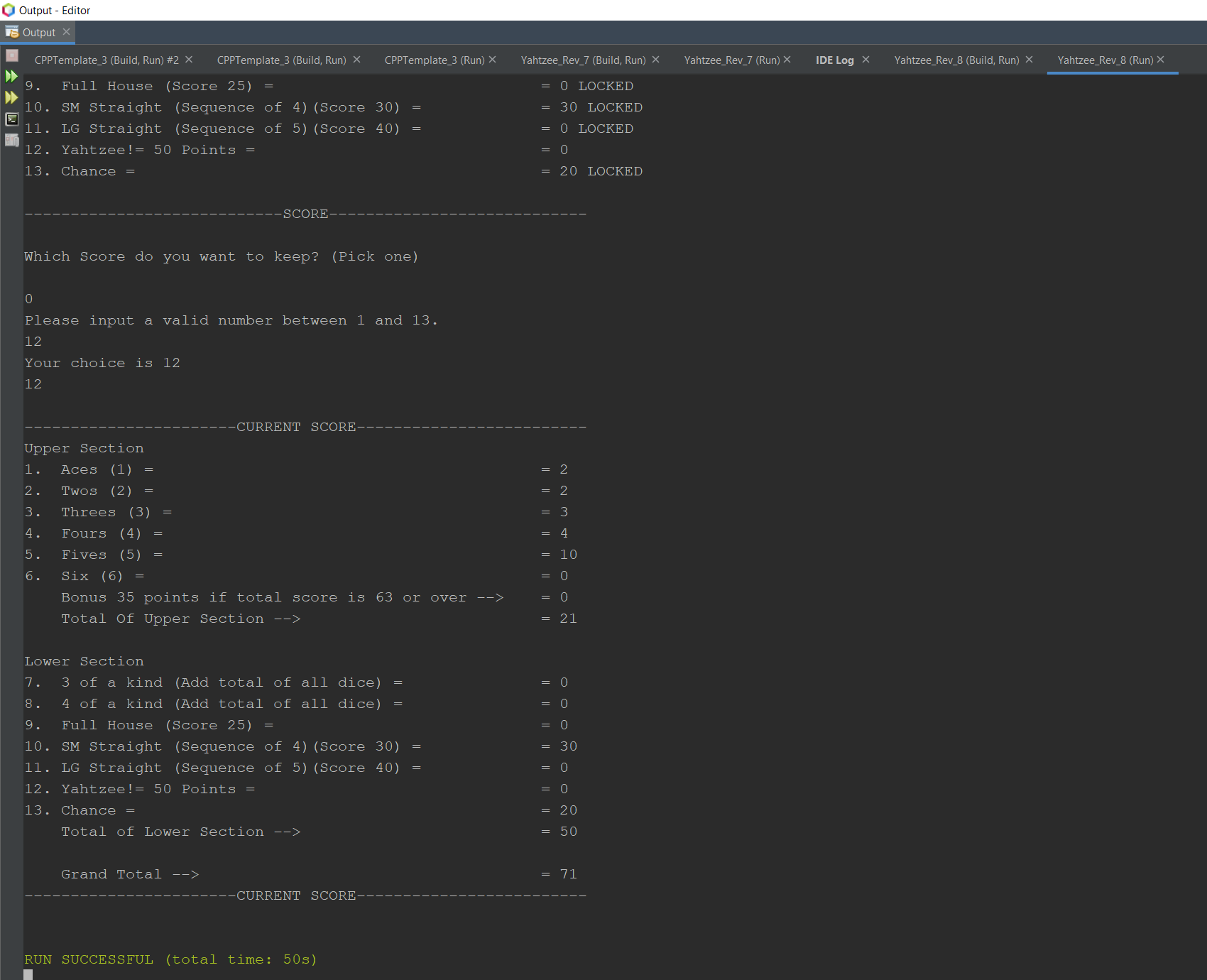








The code continues for a while. I skipped to the end here.



**9. References**

1. Dr. Lehr’s Lectures & Lab

2. “Starting Out with C++: From Control Structures through Objects” Gaddis, Tony. 8th Edition. (Textbook)

**10.Program**

/\*

\* File: main.cpp

\* Author: Jessriel Menguito

\* Created on July 25, 2022, 6:23 PM

\* Purpose: Project 1 - Yahtzee

\*/

//System Libraries

#include <iostream>

#include <iomanip>

#include <cmath>

#include <cstdlib>

#include <fstream>

#include <string>

#include <ctime>

using namespace std;

//User Libraries

//Global Constants

//Mathematical/Physics/Conversions, Higher dimensioned arrays

//Function Prototypes

void dice(int,int,int,int,int,bool,bool,bool,bool,bool);

void dice2(int,int,int,int,int,bool,bool,bool,bool,bool);

void drwDie1();

void drwDie2();

void drwDie3();

void drwDie4();

void drwDie5();

void drwDie6();

void pickDie(int);

int rmvDup(int arr[], int n){

int temp[n];

int j = 0;

// If current element is not equal to next element

// then store that current element

for (int i = 0; i < n - 1; i++)

if (arr[i] != arr[i + 1])

temp[j++] = arr[i];

// Store the last element as whether it is unique or

// repeated, it hasn't stored previously

temp[j++] = arr[n - 1];

// Modify original array

for (int i = 0; i < j; i++)

arr[i] = temp[i];

return j;

}

//Execution Begins Here

int main(int argc, char\*\* argv) {

//Initialize the Random Number Seed

srand(time(0));

//Declare Variables

float i;

int d1,d2,d3,d4,d5,loop=0,fin,choice,pick,points;

int t1=0,t2=0,t3=0,t4=0,t5=0,t6=0,t7=0,t8=0,t9=0,t10=0,t11=0,t12=0,t14;

string title;

char yn,y,n;

//Initialize Variables

bool o1=true,

o2=true,

o3=true,

o4=true,

o5=true,

s1=true,

s2=true,

s3=true,

s4=true,

s5=true,

s6=true,

s7=true,

s8=true,

s9=true,

s10=true,

s11=true,

s12=true,

s13=true;

const int game=1,

rules=2,

turn=3, //How many turns

die=6; //How many sides on a die

int count2=0, //Starting number for round

smstr=0,

lgstr=0,

Yahtzee=0,

countY=0,

bonus=0,

count3=1, //Starting number for number of games

roundY=13, //How many games

games=20;

//Array zeroing

int score[games],

player1[games],

player2[games];

for (int count=0;count<games;count++){

score[count]=0;

player1[count]=0;

player2[count]=0;

}

title="Let's Play Yahtzee!";

//Map inputs to outputs -> The Process

// Output Title

cout<<endl<<title<<endl<<endl;

//Output Yahtzee Score to text file

ofstream outputFile;

outputFile.open("YahtzeeScore.txt");

//First Roll

d1=rand()%die+1;

d2=rand()%die+1;

d3=rand()%die+1;

d4=rand()%die+1;

d5=rand()%die+1;

//Game Loop

do{

cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~ROUND "<<count3<<"~~~~~~~~~~~~~~~~~~~~~~~~~~"<<endl;

//Reset score

for (int count=0;count<games;count++){

score[count]=0;

//Reset yes no

yn=y;

//Reset Holds

o1=true;

o2=true;

o3=true;

o4=true;

o5=true;

//Reset count

count2=0;

}

//Turn Loop.

do{

//Roll dice that have bool true

int die=6;

if (o1==1){

d1=rand()%die+1;

}

if (o2==1){

d2=rand()%die+1;

}

if (o3==1){

d3=rand()%die+1;

}

if (o4==1){

d4=rand()%die+1;

}

if (o5==1){

d5=rand()%die+1;

}

//Output dice that are not set aside

dice(d1,d2,d3,d4,d5,o1,o2,o3,o4,o5);

if (count2+1<turn){

cout<<endl<<"How many dice do you want to set aside?"<<endl;

//Verify valid input

while((!(cin>>choice))||(choice<0||choice>5)){

//Error

cout<<"Please input a valid number between 0 and 5."<<endl;

//Clear the previous error

cin.clear();

//Discard value

cin.ignore(123, '\n');

}

//Which dice do you want to set aside?

for (loop=0;loop<choice;loop++){ //Loop the dice to remove

cout<<"Which dice do you want to set aside?"<<endl<<endl;

dice(d1,d2,d3,d4,d5,o1,o2,o3,o4,o5);//Call dice bank

cout<<endl;

//Verify valid input for which die the player will pick

while((!(cin>>pick))||(pick<0||pick>5)){

//Error

cout<<"Please input a valid number between 0 and 5."<<endl;

//Clear the previous error

cin.clear();

//Discard value

cin.ignore(123, '\n');

cout<<pick<<endl;

}

if (pick==1){//Die 1

o1=false;

}

if (pick==2){//Die 2

o2=false;

}

if (pick==3){//Die 3

o3=false;

}

if (pick==4){//Die 4

o4=false;

}

if (pick==5){//Die 5

o5=false;

}

}

dice2(d1,d2,d3,d4,d5,o1,o2,o3,o4,o5); //Call dice bank 2 to show what was set aside

}

//The player can end their turn here or continue rolling

count2++;

if (count2<turn){

cout<<endl<<"Reroll?"<<endl<<endl;

cout<<"Type n for NO and any key for YES."<<endl<<endl; //Ask the user if they want a reroll

cin>>yn;

if (yn=='n'||yn=='N'){

count2=turn;

}

}

}while (count2<turn);

/\*

d1=2;

d2=2;

d3=2;

d4=2;

d5=2;

\*/

cout<<endl<<"Your final dice numbers are:"<<endl;

dice(d1,d2,d3,d4,d5,o1,o2,o3,o4,o5);

//Scoring Below

{

//Sort Dice

int number[4],a;

//Set value of each die across array

number[0]=d1;

number[1]=d2;

number[2]=d3;

number[3]=d4;

number[4]=d5;

for(int j=0; j<5; j++){

for(int k=0; k<4; k++){

if (number[k]>number[k+1]){

a=number[k];

number[k]=number[k+1];

number[k+1]=a;

}

}

}

/\*

for(int count=0;count<5;count++){

cout<<"number="<<number[count]<<endl;

}

\*/

//Reset scores to 0

t1=0;

t2=0;

t3=0;

t4=0;

t5=0;

t6=0;

t8=0;

t9=0;

t10=0;

smstr=0;

lgstr=0;

Yahtzee=0;

t7=0;

bonus=0;

//Check to see if number is equal to one. Output t1

if (d1==1||d2==1||d3==1||d4==1||d5==1){

for (int count=0;count<5;count++){

if (number[count]==1){

t1+=1;

}

}

}

//check to see if number is equal to two. Output t2

if (d1==2||d2==2||d3==2||d4==2||d5==2){

for (int count=0;count<5;count++){

if (number[count]==2){

t2+=2;

}

}

}

//check to see if the number is equal to three. Output t3

if (d1==3||d2==3||d3==3||d4==3||d5==3){

for (int count=0;count<5;count++){

if (number[count]==3){

t3+=3;

}

}

}

//check to see if the number is equal to four. Output t4

if (d1==4||d2==4||d3==4||d4==4||d5==4){

for (int count=0;count<5;count++){

if (number[count]==4){

t4+=4;

}

}

}

//check to see if the number is equal to five. Output t5

if (d1==5||d2==5||d3==5||d4==5||d5==5){

for (int count=0;count<5;count++){

if (number[count]==5){

t5+=5;

}

}

}

//check to see if the number is equal to six. Output t6

if (d1==6||d2==6||d3==6||d4==6||d5==6){

for (int count=0;count<5;count++){

if (number[count]==6){

t6+=6;

}

}

}

//check for three of a kind. Output t8

if (d1==d2&&d2==d3){//check triplet 1

t8=d1+d2+d3;

}

if (d1==d2&&d2==d4){//check triplet 2

t8=d1+d2+d4;

}

if (d1==d2&&d2==d5){//check triplet 3

t8=d1+d2+d5;

}

if (d2==d3&&d3==d4){//check triplet 4

t8=d2+d3+d4;

}

if (d2==d3&&d3==d5){//check triplet 5

t8=d2+d3+d5;

}

if (d1==d4&&d4==d5){//check triplet 6

t8=d1+d4+d5;

}

if (d2==d4&&d4==d5){//check triplet 7

t8=d2+d4+d5;

}

if (d1==d3&&d3==d5){

t8=d1+d3+d5;

}

if (d1==d2&&d2==d5){

t8=d1+d2+d5;

}

if (d3==d4&&d4==d5){

t8=d3+d4+d5;

}

//check for four of a kind. Output t9

if (d1==d2&&d2==d3&&d2==d3&&d3==d4){//check quad 1

t9=d1+d2+d3+d4;

}

else if (d1==d2&&d2==d3&&d3==d5){//check quad 2

t9=d1+d2+d3+d5;

}

else if (d1==d2&&d2==d4&&d4==d5){//check quad 3

t9=d1+d2+d4+d5;

}

else if (d1==d3&&d3==d4&&d4==d5){//check quad 4

t9=d1+d3+d4+d5;

}

else if (d2==d3&&d3==d4&&d4==d5){//check quad 5

t9=d2+d3+d4+d5;

}

else

t9=0;

//Check for a house. Check triplet then check for pair. Output t10

if (d1==d2&&d2==d3){

if (d4==d5){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d1==d2&&d2==d4){

if (d3==d5){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d1==d2&&d2==d5){

if (d4==d3){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d2==d3&&d3==d4){

if (d1==d5){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d2==d3&&d3==d5){

if (d4==d1){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d1==d4&&d4==d5){

if (d2==d3){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d2==d4&&d4==d5){

if (d1==d3){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d1==d3&&d3==d5){

if (d2==d4){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d1==d2&&d2==d5){

if (d3==d4){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

else if (d3==d4&&d4==d5){

if (d1==d2){//Check if there is a pair with the remaining

t10=25;

}

else t10=0;

}

t10=0;

int n = 5;

// rmvDup() returns new size of array.

n = rmvDup(number, n);

/\*

// Print updated array

for (int i = 0; i < n; i++)

cout << "unique="<<number[i] <<endl;

\*/

//Check for small straight. Output smstr

if(number[0]==1&&number[1]==2&&number[2]==3&&number[3]==4)

smstr=30;

if(number[0]==2&&number[1]==3&&number[2]==4&&number[3]==5)

smstr=30;

if(number[0]==3&&number[1]==4&&number[2]==5&&number[3]==6)

smstr=30;

//Check for large straight. Sum of 1-5=15. Output lgstr

if(number[0]==1&&number[1]==2&&number[2]==3&&number[3]==4&&number[4]==5)

lgstr=40;

if(number[0]==2&&number[1]==3&&number[2]==4&&number[3]==5&&number[4]==6)

lgstr=40;

//Check for Yahtzee. Output Yahtzee

if (d1==d2&&d2==d3&&d3==d4&&d4==d5){//If all of the numbers are the same

Yahtzee=50;

if (countY>0){

Yahtzee=50+(countY\*100);

}

}

//Sum of dice totals for Chance

t14=d1+d2+d3+d4+d5;

}//Scoring Above

//Display Results

cout<<fixed<<setw(54)<<right<<endl<<"----------------------------SCORE----------------------------"<<endl;

cout<<"Upper Section"<<endl;

cout<<fixed<<setw(54)<<left<< "1. Aces (1) = "<<setw(4)<<right<<"= ";

if (s1==true){

cout<<t1<<endl;

}

else{

cout<<player1[0]<<" LOCKED"<<endl;

};

cout<<fixed<<setw(54)<<left<< "2. Twos (2) = "<<setw(4)<<right<<"= ";

if (s2==true){

cout<<t2<<endl;

}

else{

cout<<player1[1]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "3. Threes (3) = "<<setw(4)<<right<<"= ";

if (s3==true){

cout<< t3<<endl;

}

else{

cout<<player1[2]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "4. Fours (4) = "<<setw(4)<<right<<"= ";

if (s4==true){

cout<<t4<<endl;

}

else{

cout<<player1[3]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "5. Fives (5) = "<<setw(4)<<right<<"= ";

if (s5==true){

cout<<t5<<endl;

}

else{

cout<<player1[4]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "6. Six (6) = "<<setw(4)<<right<<"= ";

if (s6==true){

cout<<t6<<endl;

}

else{

cout<<player1[5]<<" LOCKED"<<endl;

}

cout<<endl<<"Lower Section"<<endl;

cout<<fixed<<setw(54)<<left<< "7. 3 of a kind (Add total of all dice) = "<<setw(4)<<right<<"= ";

if (s7==true){

cout<<t8<<endl;

}

else{

cout<<player1[8]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "8. 4 of a kind (Add total of all dice) = "<<setw(4)<<right<<"= ";

if (s8==true){

cout<<t9<<endl;

}

else{

cout<<player1[9]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "9. Full House (Score 25) = "<<setw(4)<<right<<"= ";

if (s9==true){

cout<<t10<<endl;

}

else{

cout<<player1[10]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "10. SM Straight (Sequence of 4)(Score 30) = "<<setw(4)<<right<<"= ";

if (s10==true){

cout<<smstr<<endl;

}

else{

cout<<player1[11]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "11. LG Straight (Sequence of 5)(Score 40) = "<<setw(4)<<right<<"= ";

if (s11==true){

cout<<lgstr<<endl;

}

else{

cout<<player1[12]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<< "12. Yahtzee!= 50 Points = "<<setw(4)<<right<<"= ";

if (s12==true){

cout<<Yahtzee<<endl;

}

else{

cout<<"100 x"<<countY<<endl;

}

cout<<fixed<<setw(54)<<left<< "13. Chance = "<<setw(4)<<right<<"= ";

if (s13==true){

cout<<t14<<endl;

}

else{

cout<<player1[14]<<" LOCKED"<<endl;

}

cout<<fixed<<setw(54)<<left<<endl<< "----------------------------SCORE----------------------------"<<endl<<endl;

//Set results to temporary array

score[0]= t1; //Aces

score[1]= t2; //Twos

score[2]= t3; //Threes

score[3]= t4; //Fours

score[4]= t5; //Fives

score[5]= t6; //Sixes

score[6]= bonus; //Bonus 35 if over 63 above

score[7]= t12; //Total of upper section

score[8]= t8; //3 of a kind

score[9]= t9; //4 of a kind

score[10]= t10; //Full house

score[11]= smstr; //Small Straight

score[12]= lgstr; //Large Straight

score[13]= Yahtzee; //Yahtzee

score[14]= t14; //Chance

score[15]= t11; //Total of lower section

score[16]= fin; //Grand total

//Player picks which score to keep

cout<<"Which Score do you want to keep? (Pick one)"<<endl<<endl;

//Verify input

while((!(cin>>points))||(points<1||points>13)){

//Error

cout<<"Please input a valid number between 1 and 13."<<endl;

//Clear the previous error

cin.clear();

//Discard value

cin.ignore(123, '\n');

}

cout<<"Your choice is "<<points<<endl;

if (points==12){

countY++;

}

cout<<points<<endl;

//Send players points to array to save for next game. Bool to lock score.

if (points==1){

player1[0]=score[0];

s1=false;

}

else if (points==2){

player1[1]=score[1];

s2=false;

}

else if (points==3){

player1[2]=score[2];

s3=false;

}

else if (points==4){

player1[3]=score[3];

s4=false;

}

else if (points==5){

player1[4]=score[4];

s5=false;

}

else if (points==6){

player1[5]=score[5];

s6=false;

}

else if (points==7){

player1[8]=score[8];

s7=false;

}

else if (points==8){

player1[9]=score[9];

s8=false;

}

else if (points==9){

player1[10]=score[10];

s9=false;

}

else if (points==10){

player1[11]=score[11];

s10=false;

}

else if (points==11){

player1[12]=score[12];

s11=false;

}

else if (points==12){

player1[13]=score[13];

s12=false;

}

else if (points==13){

player1[14]=score[14];

s13=false;

}

else

cout<<"error"<<endl;

//Check to see if player gets bonus points

for(int count=0;count<6;count++){

t7+=score[count];

}

if (t7>63){

player1[6]=35;

}

//Total for upper section

player1 [7]=0;

for(int count=0;count<=6;count++){

player1[7]+=player1[count];

}

//Total for lower section

player1 [15]=0;

for(int count=0;(count+8)<15;count++){

player1[15]+=player1[8+count];

}

//Grand Total

player1[16]=player1[7]+player1[15];

cout<<fixed<<setw(54)<<right<<endl<<"-----------------------CURRENT SCORE-------------------------"<<endl;

cout<<"Upper Section"<<endl;

cout<<fixed<<setw(54)<<left<< "1. Aces (1) = "<<setw(4)<<right<<"= " << player1[0]<<endl;

cout<<fixed<<setw(54)<<left<< "2. Twos (2) = "<<setw(4)<<right<<"= "<< player1[1]<<endl;

cout<<fixed<<setw(54)<<left<< "3. Threes (3) = "<<setw(4)<<right<<"= "<< player1[2]<<endl;

cout<<fixed<<setw(54)<<left<< "4. Fours (4) = "<<setw(4)<<right<<"= "<< player1[3]<<endl;

cout<<fixed<<setw(54)<<left<< "5. Fives (5) = "<<setw(4)<<right<<"= "<< player1[4]<<endl;

cout<<fixed<<setw(54)<<left<< "6. Six (6) = "<<setw(4)<<right<<"= "<< player1[5]<<endl;

cout<<fixed<<setw(54)<<left<< " Bonus 35 points if total score is 63 or over -->"<<setw(4)<<right<<"= "<<player1[6]<<endl;

cout<<fixed<<setw(54)<<left<< " Total Of Upper Section --> "<<setw(4)<<right<<"= "<< player1[7]<<endl;

cout<<endl<<"Lower Section"<<endl;

cout<<fixed<<setw(54)<<left<< "7. 3 of a kind (Add total of all dice) = "<<setw(4)<<right<<"= "<< player1[8]<<endl;

cout<<fixed<<setw(54)<<left<< "8. 4 of a kind (Add total of all dice) = "<<setw(4)<<right<<"= "<< player1[9]<<endl;

cout<<fixed<<setw(54)<<left<< "9. Full House (Score 25) = "<<setw(4)<<right<<"= "<< player1[10]<<endl;

cout<<fixed<<setw(54)<<left<< "10. SM Straight (Sequence of 4)(Score 30) = "<<setw(4)<<right<<"= "<< player1[11]<<endl;

cout<<fixed<<setw(54)<<left<< "11. LG Straight (Sequence of 5)(Score 40) = "<<setw(4)<<right<<"= "<< player1[12]<<endl;

cout<<fixed<<setw(54)<<left<< "12. Yahtzee!= 50 Points = "<<setw(4)<<right<<"= "<< player1[13]<<endl;

cout<<fixed<<setw(54)<<left<< "13. Chance = "<<setw(4)<<right<<"= "<< player1[14]<<endl;

cout<<fixed<<setw(54)<<left<< " Total of Lower Section --> "<<setw(4)<<right<<"= "<< player1[15];

cout<<fixed<<setw(54)<<left<<endl<<endl<<" Grand Total --> "<<setw(4)<<right<<"= "<< player1[16];

cout<<fixed<<setw(54)<<left<<endl<< "-----------------------CURRENT SCORE-------------------------"<<endl<<endl;

count3++;

}while (count3<=roundY);

//Close the file

outputFile.close();

//Exit stage right

return 0;

}

void dice(int d1,int d2,int d3,int d4,int d5,bool o1,bool o2,bool o3,bool o4,bool o5){

if (o1==true){

pickDie(d1);

cout << "Die 1 = "<<d1<<endl;

}

else{

pickDie(d1);

cout << "Die 1 = "<<d1<<" HOLDING"<<endl;

}

if (o2==true){

pickDie(d2);

cout << "Die 2 = "<<d2<<endl;

}

else{

pickDie(d2);

cout << "Die 2 = "<<d2<<" HOLDING"<<endl;

}

if (o3==true){

pickDie(d3);

cout << "Die 3 = "<<d3<<endl;

}

else{

pickDie(d3);

cout << "Die 3 = "<<d3<<" HOLDING"<<endl;

}

if (o4==true){

pickDie(d4);

cout << "Die 4 = "<<d4<<endl;

}

else{

pickDie(d4);

cout << "Die 4 = "<<d4<<" HOLDING"<<endl;

}

if (o5==true){

pickDie(d5);

cout << "Die 5 = "<<d5<<endl;

}

else{

pickDie(d5);

cout << "Die 5 = "<<d5<<" HOLDING"<<endl;

}

cout<<endl;

}

void dice2(int d1,int d2,int d3,int d4,int d5,bool o1,bool o2,bool o3,bool o4,bool o5){

cout<< "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"You have set aside dice numbers:"<<endl;

if (o1==false){

pickDie(d1);

cout << "Die 1 = "<<d1<<endl;

}

if (o2==false){

pickDie(d2);

cout << "Die 2 = "<<d2<<endl;

}

if (o3==false){

pickDie(d3);

cout << "Die 3 = "<<d3<<endl;

}

if (o4==false){

pickDie(d4);

cout << "Die 4 = "<<d4<<endl;

}

if (o5==false){

pickDie(d5);

cout << "Die 5 = "<<d5<<endl;

}

if (o5==false&&o4==false&&o3==false&&o2==false&&o1==false)

cout<< "No numbers set aside."<<endl;

cout<< "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<endl;

}

void pickDie(int x){

if (x==1){

cout << "\*\*\*\*\*\*\*\*\*\n"

<< "\* \*\n"

<< "\* O \*\n"

<< "\* \*\n"

<< "\*\*\*\*\*\*\*\*\*\n";

}

if (x==2){

cout << "\*\*\*\*\*\*\*\*\*\n"

<< "\* O \*\n"

<< "\* \*\n"

<< "\* O \*\n"

<< "\*\*\*\*\*\*\*\*\*\n";

}

if (x==3){

cout << "\*\*\*\*\*\*\*\*\*\n"

<< "\* O \*\n"

<< "\* O \*\n"

<< "\* O \*\n"

<< "\*\*\*\*\*\*\*\*\*\n";

}

if (x==4){

cout << "\*\*\*\*\*\*\*\*\*\n"

<< "\* O O \*\n"

<< "\* \*\n"

<< "\* O O \*\n"

<< "\*\*\*\*\*\*\*\*\*\n";

}

if (x==5){

cout << "\*\*\*\*\*\*\*\*\*\n"

<< "\* O O \*\n"

<< "\* O \*\n"

<< "\* O O \*\n"

<< "\*\*\*\*\*\*\*\*\*\n";

}

if (x==6){

cout << "\*\*\*\*\*\*\*\*\*\n"

<< "\* O O \*\n"

<< "\* O O \*\n"

<< "\* O O \*\n"

<< "\*\*\*\*\*\*\*\*\*\n";

}

}

