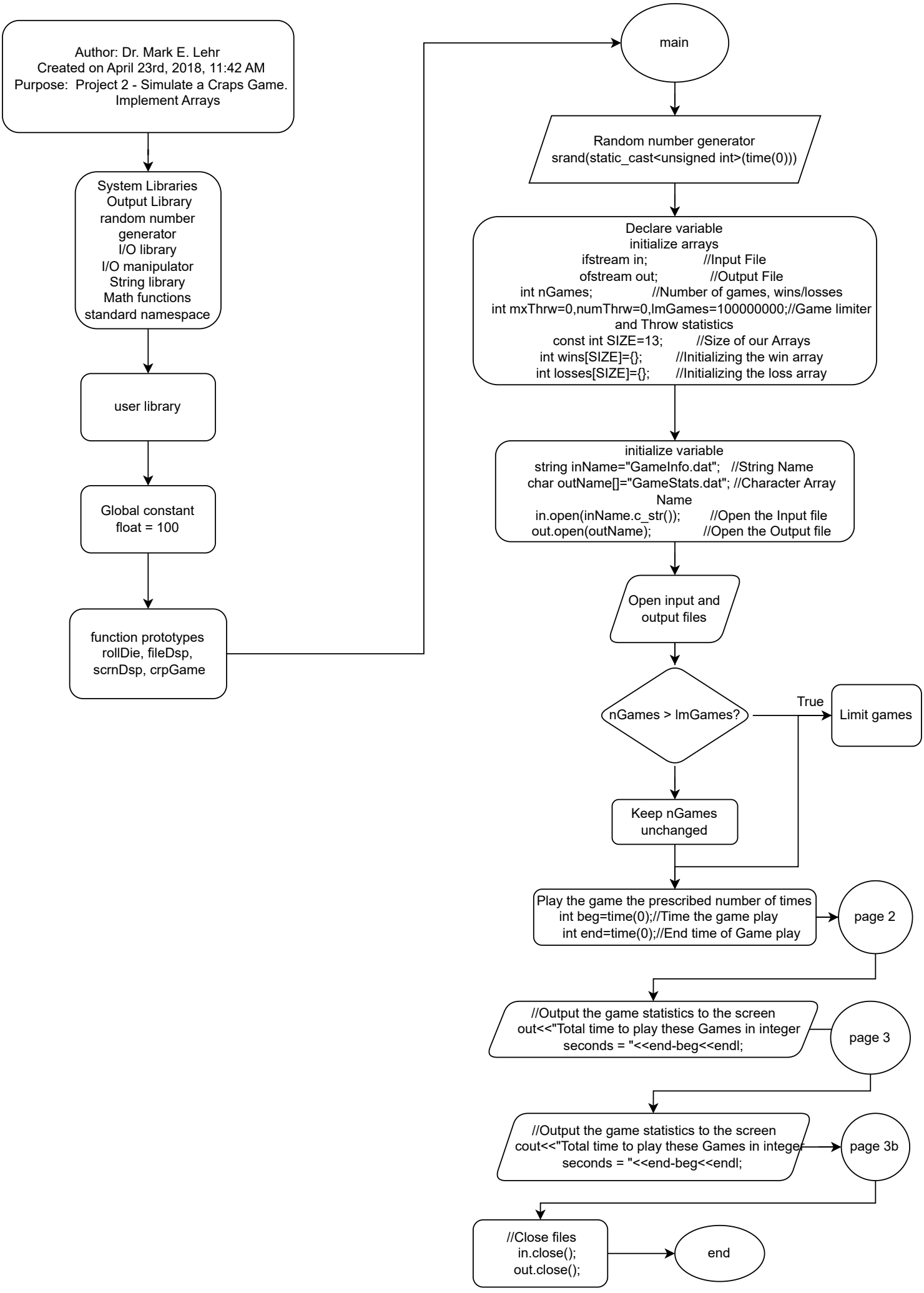
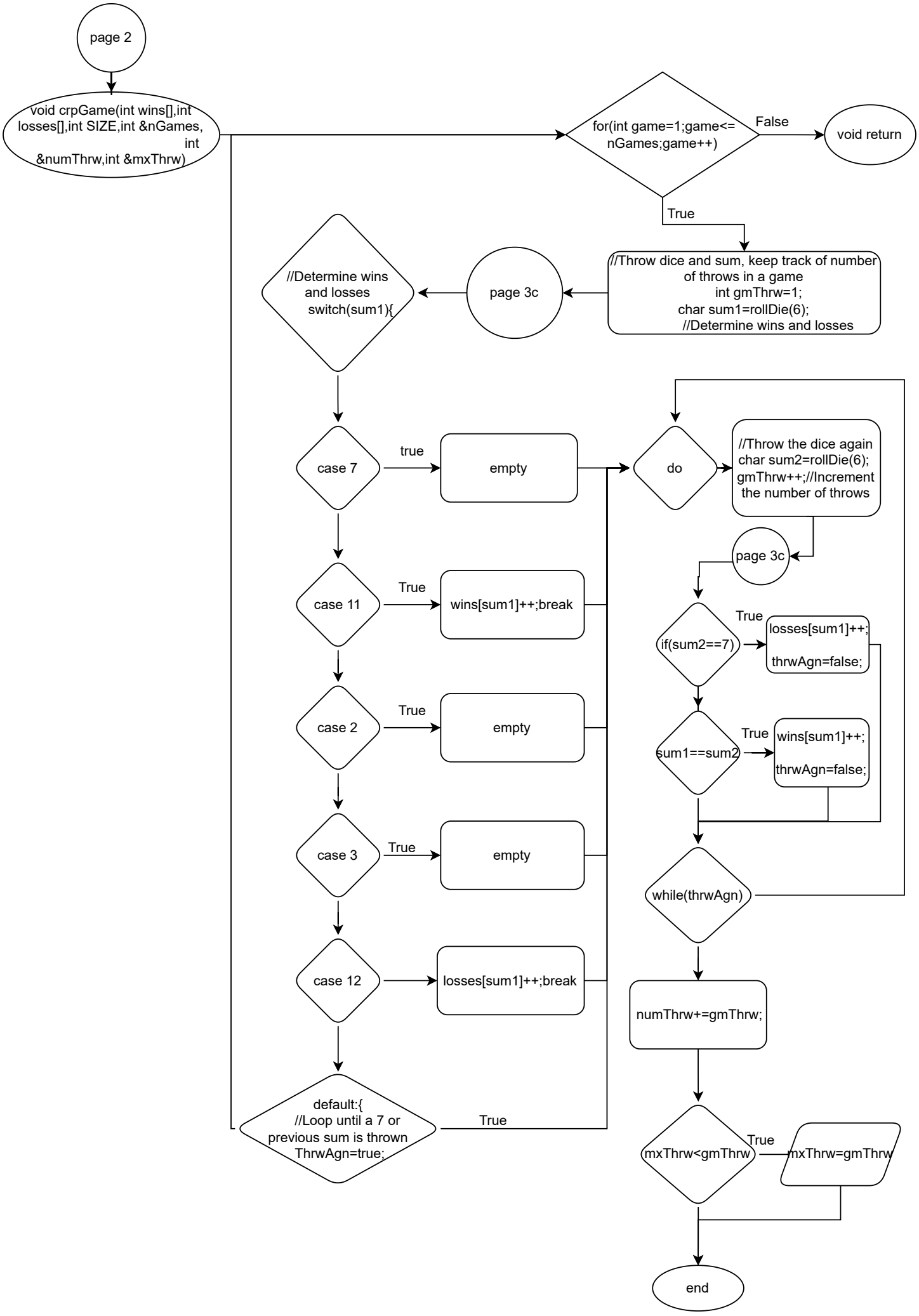
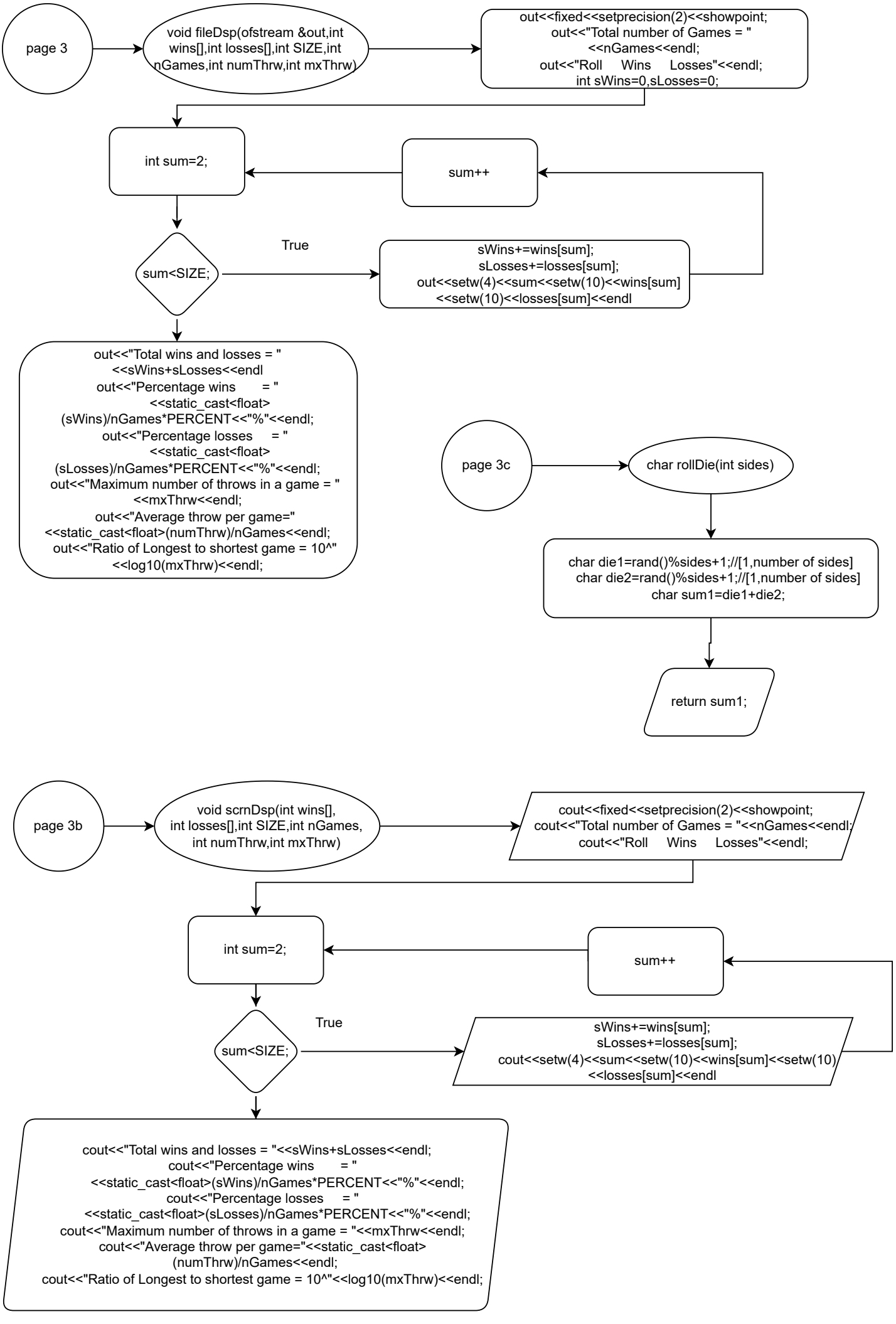

Author: Dr. Mark E. Lehr
Created on April 23rd, 2018, 11:42 AM
Purpose: Project 2 - Simulate a Craps Game.
Implement Arrays
System Libraries
Output Library
random number
generator
I/O library
I/O manipulator
String library
Math functions
standard namespace
user library
Global constant
float = 100
function prototypes
rollDie, fileDsp,
scrnDsp, crpGame
main
Random number generator
srand(static_cast<unsigned int>(time(0)))
Declare variable
initialize arrays
ifstream in; //Input File
ofstream out; //Output File
int nGames; //Number of games, wins/losses
int mxThrw=0,numThrw=0,lmGames=100000000;//Game limiter
and Throw statistics
const int SIZE=13; //Size of our Arrays
int wins[SIZE]={}; //Initializing the win array
int losses[SIZE]={}; //Initializing the loss array
initialize variable
string inName="GameInfo.dat"; //String Name
char outName[]="GameStats.dat"; //Character Array
Name
in.open(inName.c_str()); //Open the Input file
out.open(outName); //Open the Output file
Open input and
output files
nGames > lmGames?
True
Limit games
Keep nGames
unchanged
Play the game the prescribed number of times
int beg=time(0);//Time the game play
int end=time(0);//End time of Game play
page 2
//Output the game statistics to the screen
out<<"Total time to play these Games in integer
seconds = "<<end-beg<<endl;
page 3
//Output the game statistics to the screen
cout<<"Total time to play these Games in integer
seconds = "<<end-beg<<endl;
page 3b
//Close files
in.close();
out.close();
end


page 2
void crpGame(int wins[],int losses[],int SIZE,int &nGames, int &numThrw,int &mxThrw)
for(int game=1;game<= nGames;game++)
False
void return
True
//Throw dice and sum, keep track of number of throws in a game
int gmThrw=1;
char sum1=rollDie(6);
//Determine wins and losses
page 3c
//Determine wins and losses
switch(sum1){
case 7
true
empty
case 11
True
wins[sum1]++;break
case 2
True
empty
case 3
True
empty
case 12
losses[sum1]++;break
default:{
//Loop until a 7 or previous sum is thrown
ThrwAgn=true;
True
do
//Throw the dice again
char sum2=rollDie(6);
gmThrw++;//Increment the number of throws
page 3c
if(sum2==7)
True
losses[sum1]++;
thrwAgn=false;
sum1==sum2
True
wins[sum1]++;
thrwAgn=false;
while(thrwAgn)
numThrw+=gmThrw;
mxThrw<gmThrw
True
mxThrw=gmThrw
end


page 3
void fileDsp(ofstream &out,int wins[],int losses[],int SIZE,int nGames,int numThrw,int mxThrw)
out<<fixed<<setprecision(2)<<showpoint;
out<<"Total number of Games = "<<nGames<<endl;
out<<"Roll Wins Losses"<<endl;
int sWins=0,sLosses=0;
int sum=2;
sum++
sum<SIZE;
True
sWins+=wins[sum];
sLosses+=losses[sum];
out<<setw(4)<<sum<<setw(10)<<wins[sum]<<setw(10)<<losses[sum]<<endl
out<<"Total wins and losses = "<<sWins+sLosses<<endl
out<<"Percentage wins = "<<static_cast<float>(sWins)/nGames*PERCENT<<"%"<<endl;
out<<"Percentage losses = "<<static_cast<float>(sLosses)/nGames*PERCENT<<"%"<<endl;
out<<"Maximum number of throws in a game = "<<mxThrw<<endl;
out<<"Average throw per game="<<static_cast<float>(numThrw)/nGames<<endl;
out<<"Ratio of Longest to shortest game = 10^"<<log10(mxThrw)<<endl;
page 3c
char rollDie(int sides)
char die1=rand()%sides+1;//[1,number of sides]
char die2=rand()%sides+1;//[1,number of sides]
char sum1=die1+die2;
return sum1;
page 3b
void scrnDsp(int wins[],int losses[],int SIZE,int nGames,int numThrw,int mxThrw)
cout<<fixed<<setprecision(2)<<showpoint;
cout<<"Total number of Games = "<<nGames<<endl;
cout<<"Roll Wins Losses"<<endl;
int sum=2;
sum++
sum<SIZE;
True
sWins+=wins[sum];
sLosses+=losses[sum];
cout<<setw(4)<<sum<<setw(10)<<wins[sum]<<setw(10)<<losses[sum]<<endl
cout<<"Total wins and losses = "<<sWins+sLosses<<endl;
cout<<"Percentage wins = "<<static_cast<float>(sWins)/nGames*PERCENT<<"%"<<endl;
cout<<"Percentage losses = "<<static_cast<float>(sLosses)/nGames*PERCENT<<"%"<<endl;
cout<<"Maximum number of throws in a game = "<<mxThrw<<endl;
cout<<"Average throw per game="<<static_cast<float>(numThrw)/nGames<<endl;
cout<<"Ratio of Longest to shortest game = 10^"<<log10(mxThrw)<<endl;