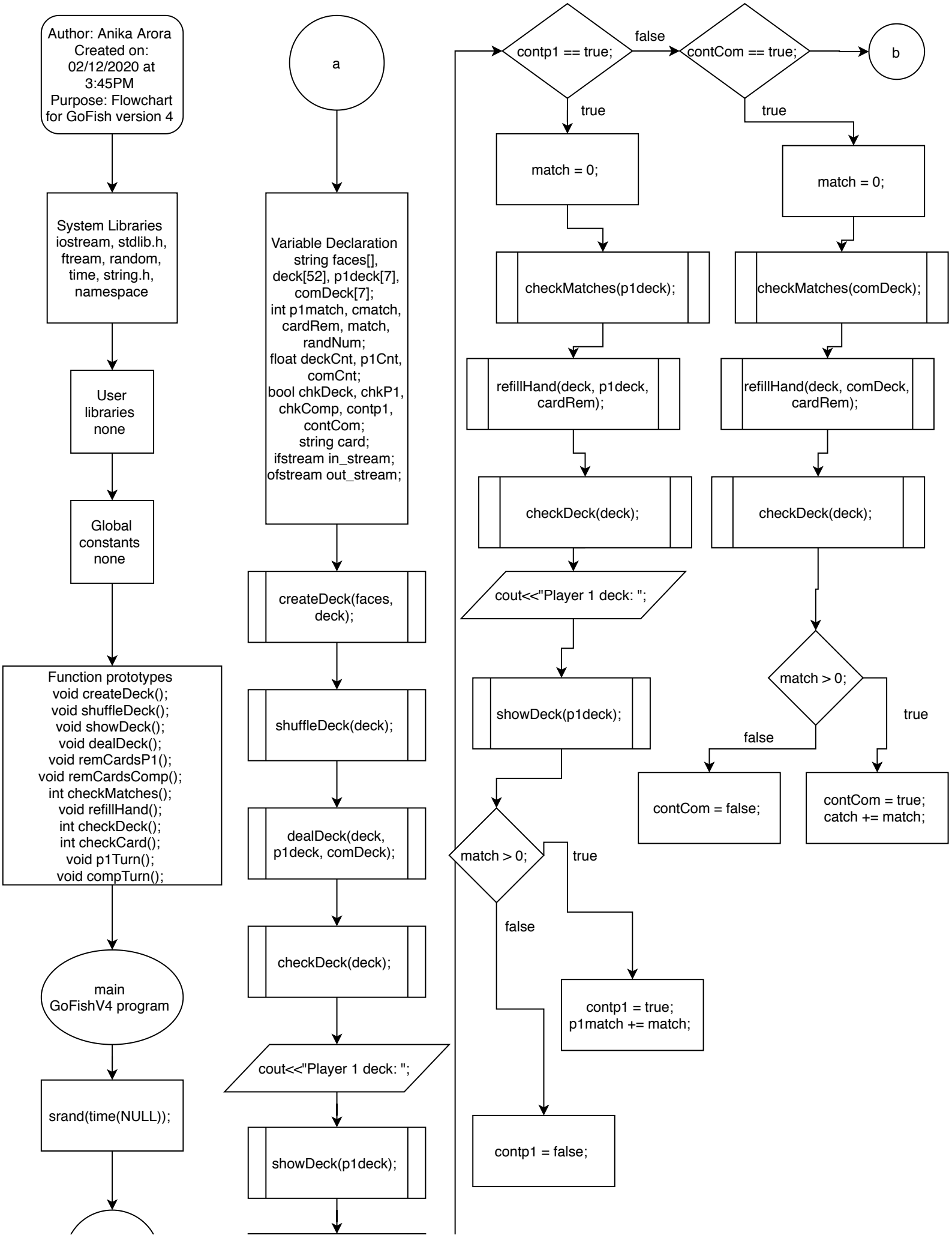
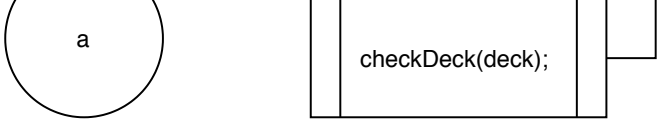
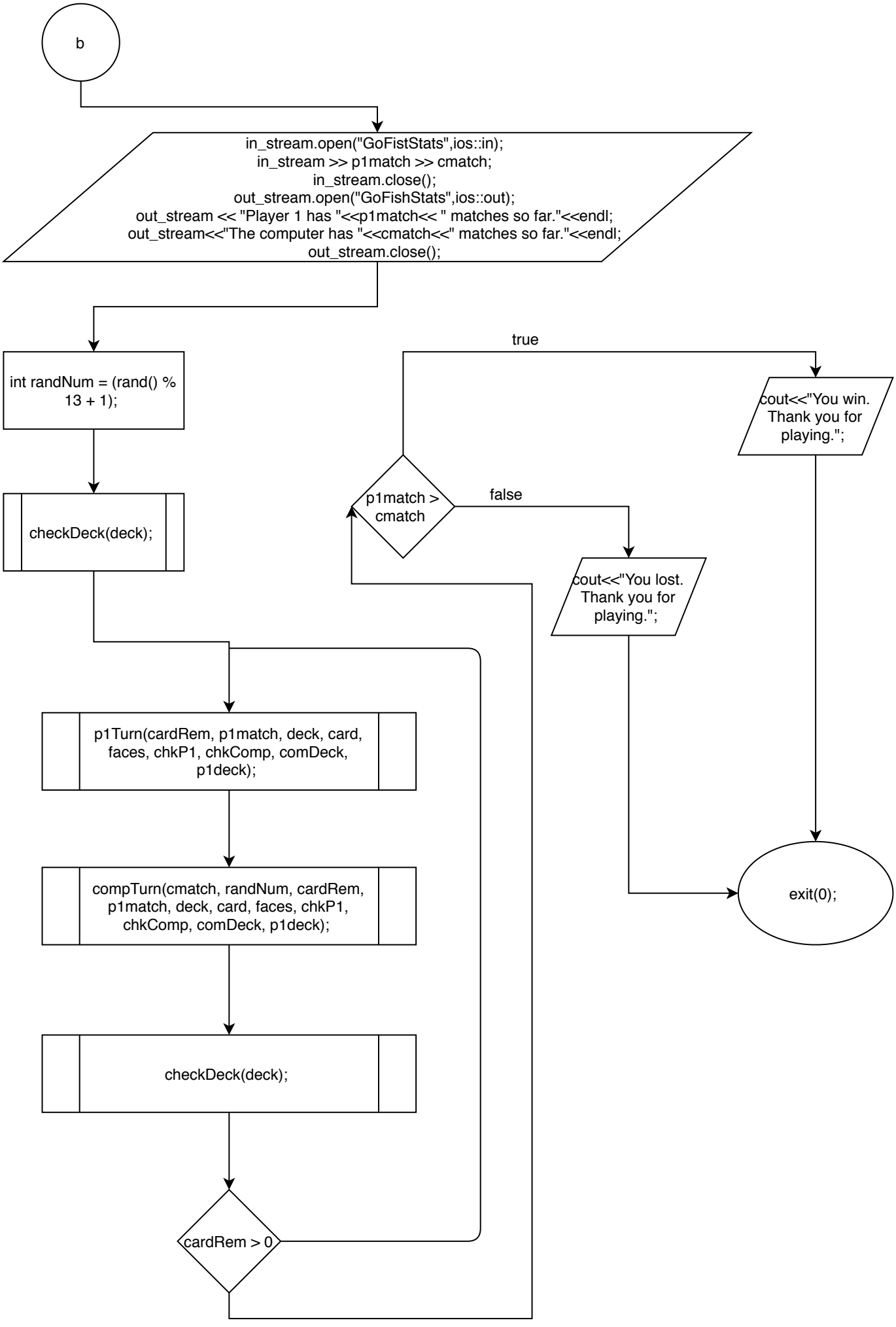
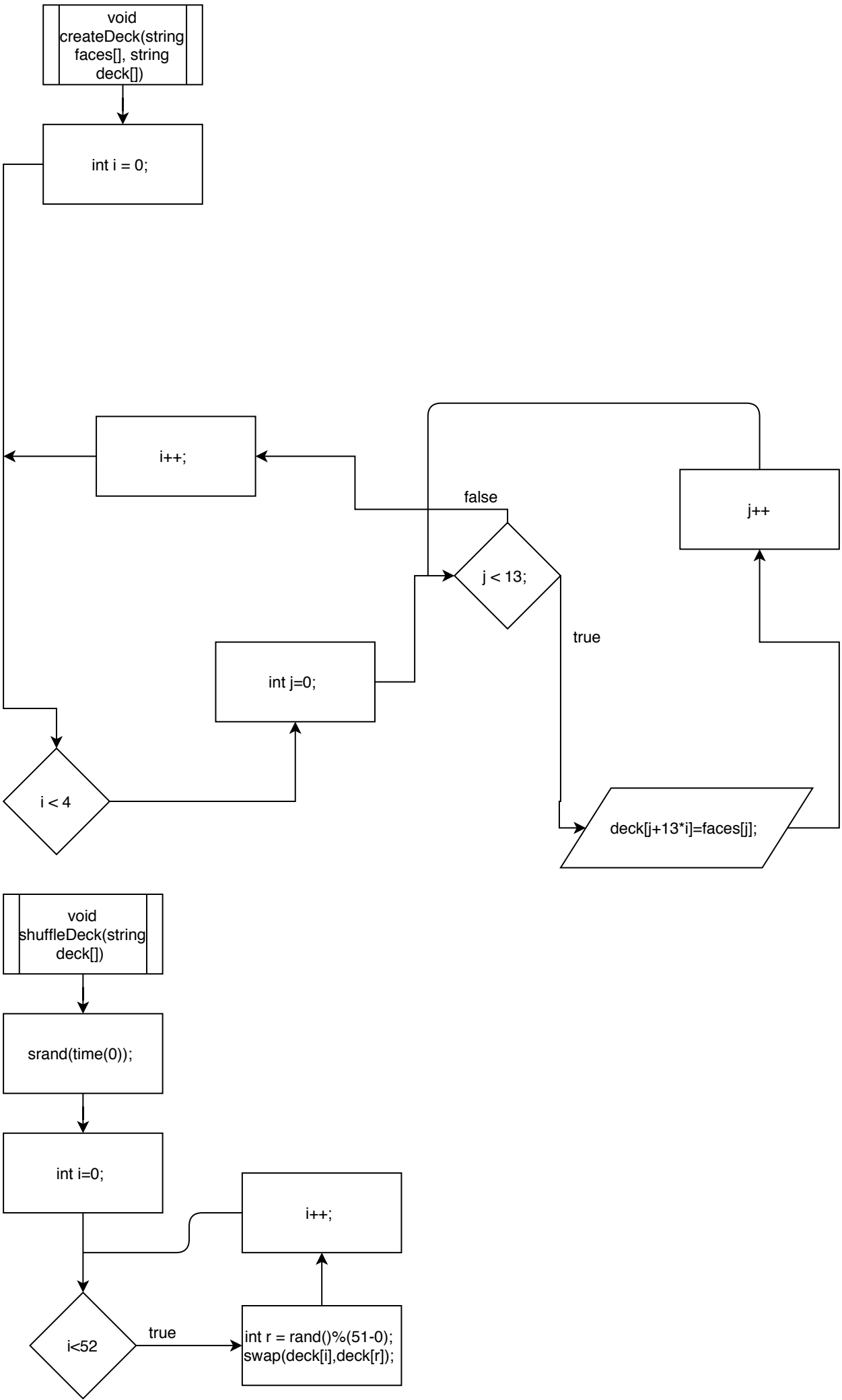
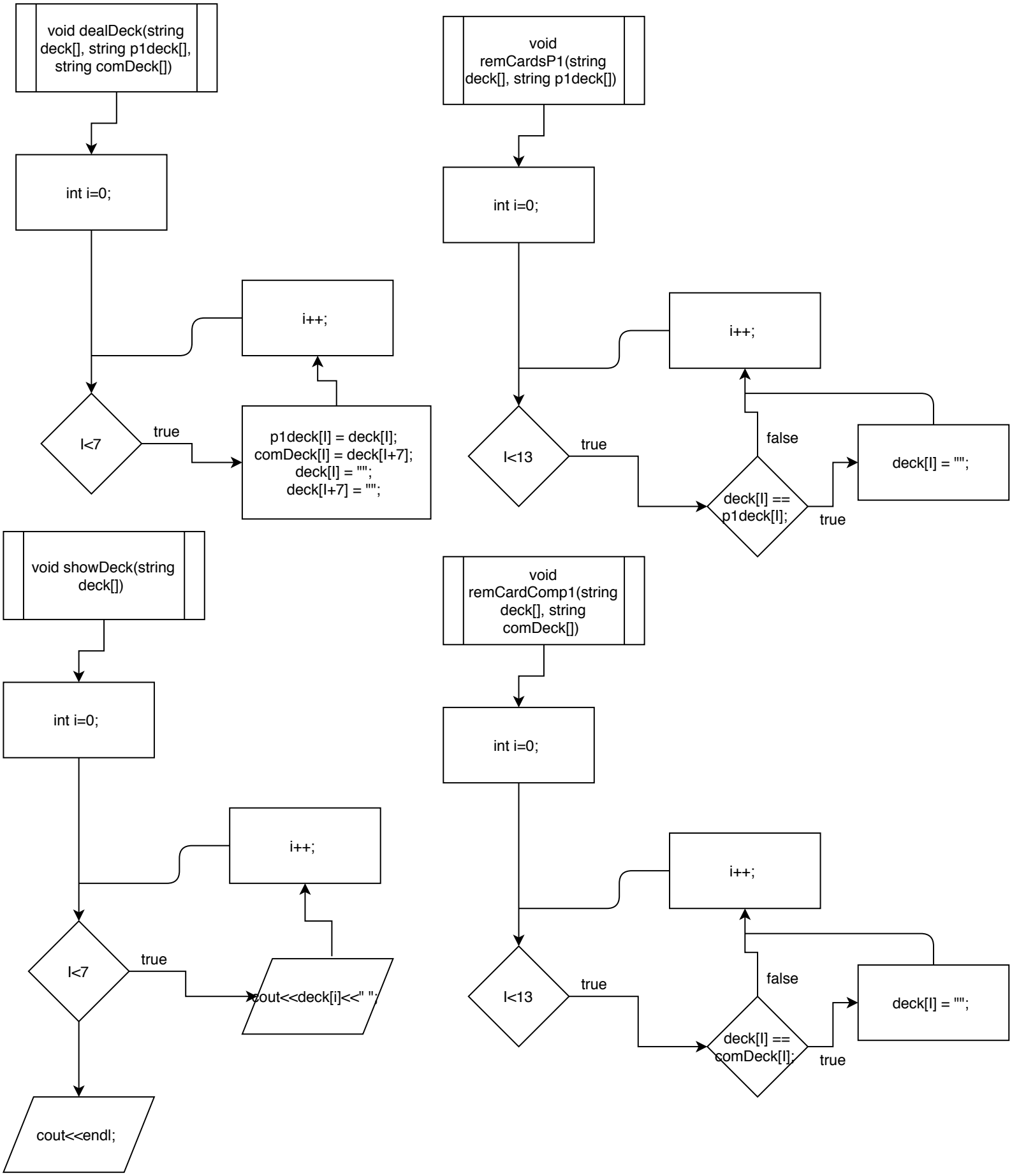
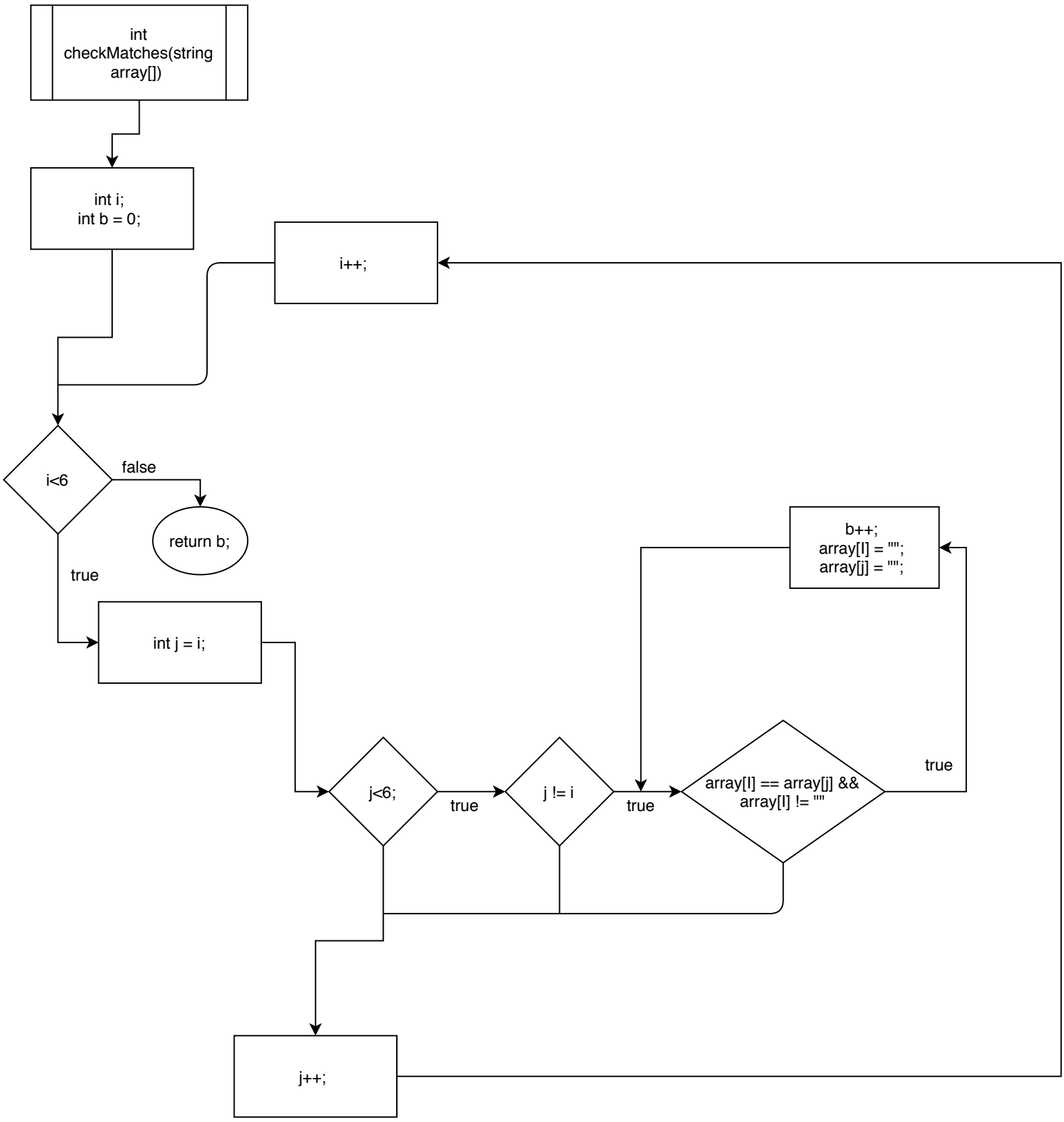

Author: Anika Arora
Created on: 02/12/2020 at 3:45PM
Purpose: Flowchart for GoFish version 4
System Libraries
iostream, stdlib.h, ftream, random, time, string.h, namespace
User libraries
none
Global constants
none
Function prototypes
void createDeck();
void shuffleDeck();
void showDeck();
void dealDeck();
void remCardsP1();
void remCardsComp();
int checkMatches();
void refillHand();
int checkDeck();
int checkCard();
void p1Turn();
void compTurn();
main
GoFishV4 program
srand(time(NULL));
a
Variable Declaration
string faces[],
deck[52], p1deck[7], comDeck[7];
int p1match, cmatch, cardRem, match, randNum;
float deckCnt, p1Cnt, comCnt;
bool chkDeck, chkP1, chkComp, contp1, contCom;
string card;
ifstream in_stream;
ofstream out_stream;
createDeck(faces, deck);
shuffleDeck(deck);
dealDeck(deck, p1deck, comDeck);
checkDeck(deck);
cout<<"Player 1 deck: ";
showDeck(p1deck);
contp1 == true;
false
contCom == true;
b
true
true
match = 0;
match = 0;
checkMatches(p1deck);
checkMatches(comDeck);
refillHand(deck, p1deck, cardRem);
refillHand(deck, comDeck, cardRem);
checkDeck(deck);
checkDeck(deck);
cout<<"Player 1 deck: ";
showDeck(p1deck);
match > 0;
true
false
contp1 = true;
p1match += match;
contp1 = false;
match > 0;
true
false
contCom = false;
contCom = true;
catch += match;

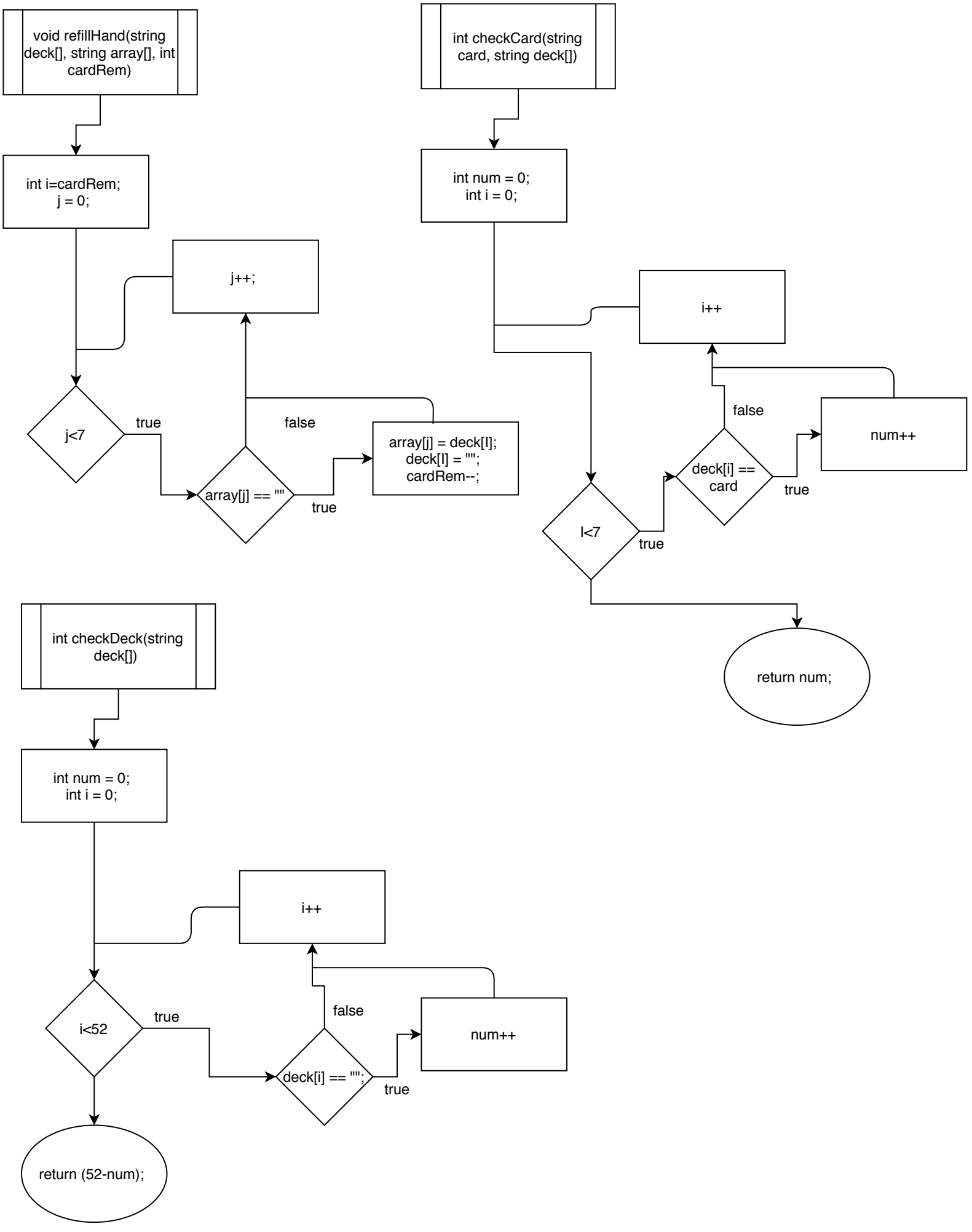

a
checkDeck(deck);

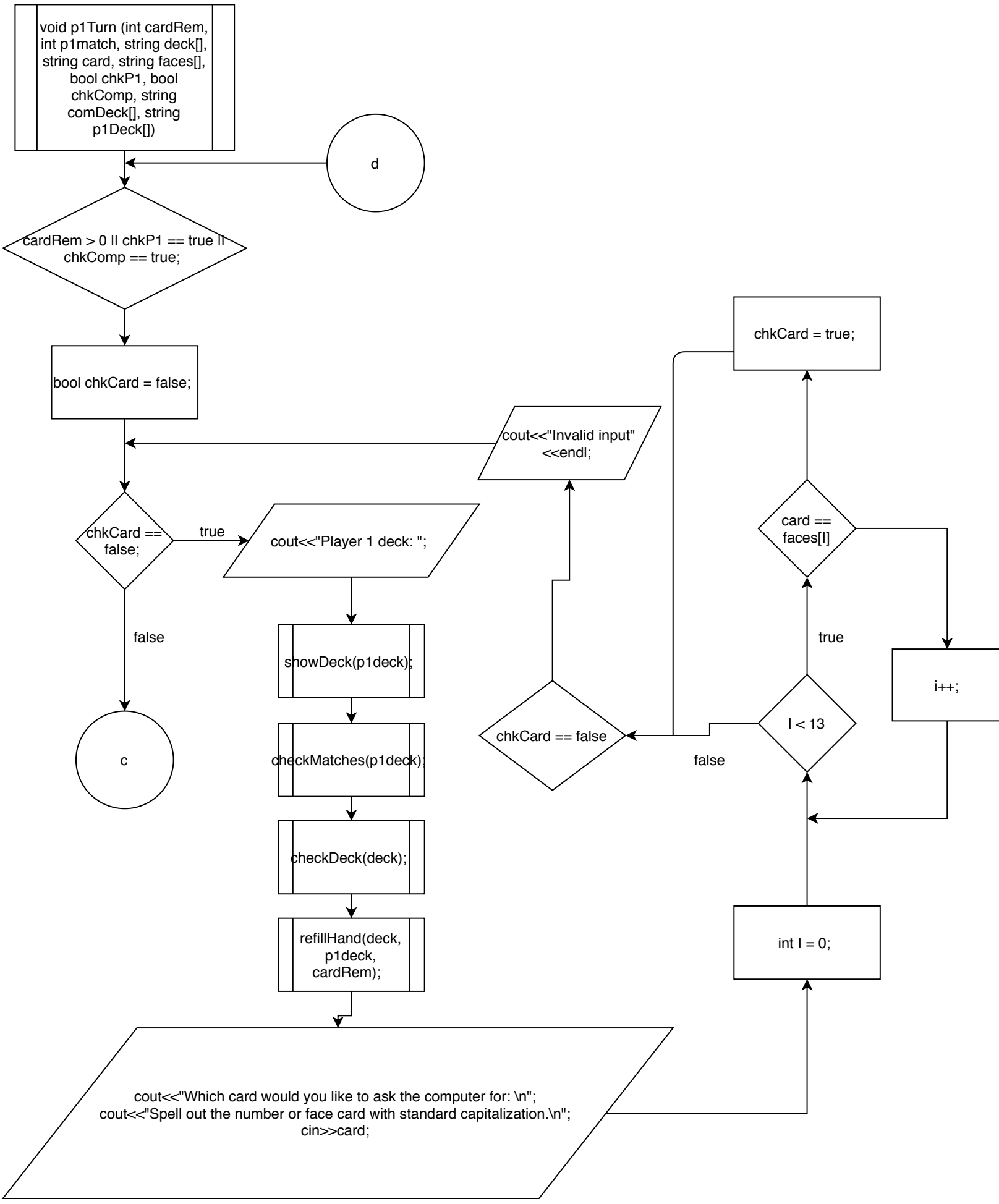

b
in_stream.open("GoFistStats",ios::in);
in_stream >> p1match >> cmatch;
in_stream.close();
out_stream.open("GoFishStats",ios::out);
out_stream << "Player 1 has "<<p1match<< " matches so far."<<endl;
out_stream<<"The computer has "<<cmatch<<" matches so far."<<endl;
out_stream.close();
int randNum = (rand() % 13 + 1);
checkDeck(deck);
p1Turn(cardRem, p1match, deck, card, faces, chkP1, chkComp, comDeck, p1deck);
compTurn(cmatch, randNum, cardRem, p1match, deck, card, faces, chkP1, chkComp, comDeck, p1deck);
checkDeck(deck);
cardRem > 0
p1match > cmatch
true
false
cout<<"You win. Thank you for playing.";
cout<<"You lost. Thank you for playing.";
exit(0);

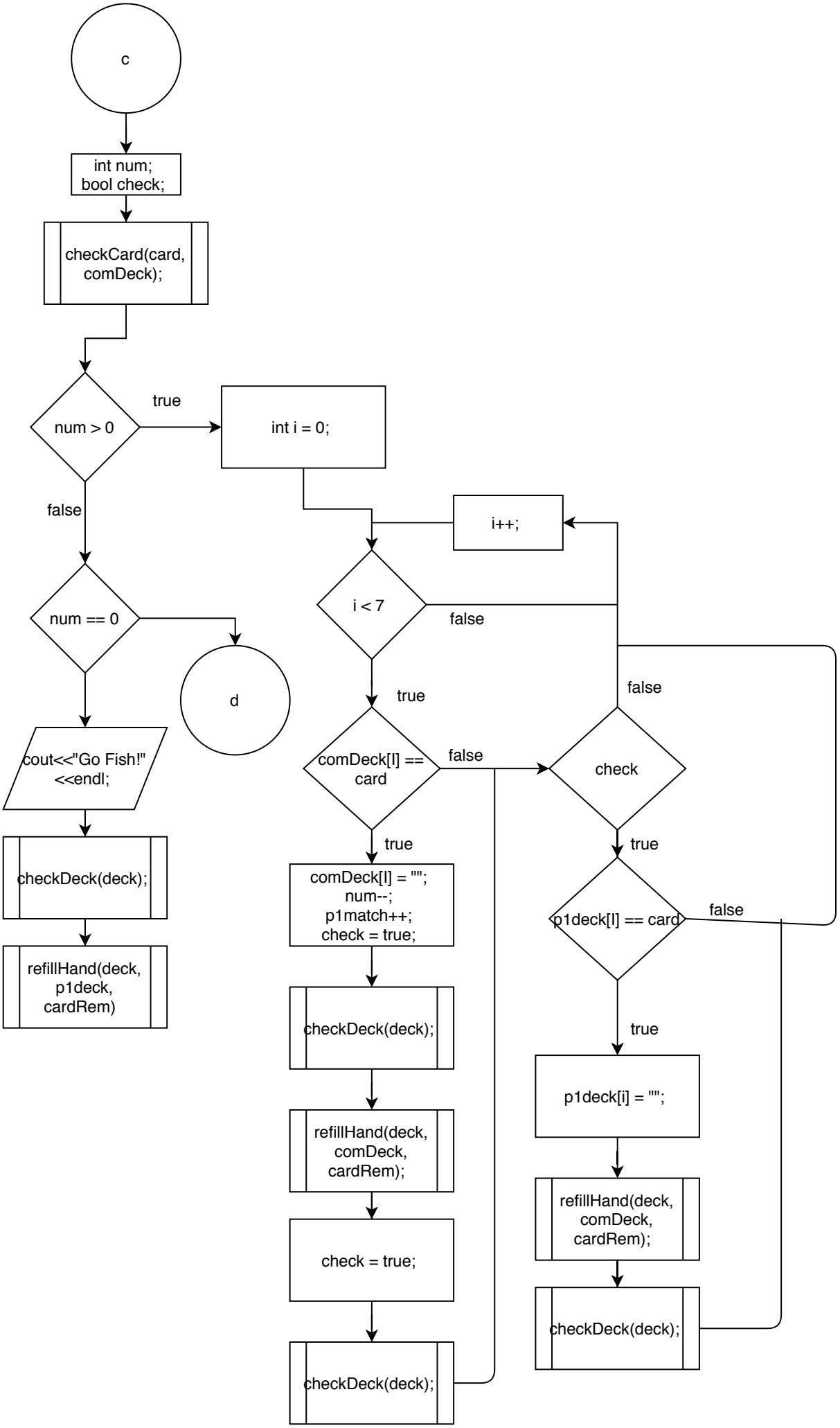

void createDeck(string faces[], string deck[])
int i = 0;
i++;
false
j++
j < 13;
true
int j=0;
i < 4
deck[j+13*i]=faces[j];
void shuffleDeck(string deck[])
srand(time(0));
int i=0;
i++;
i<52
true
int r = rand()%(51-0);
swap(deck[i],deck[r]);

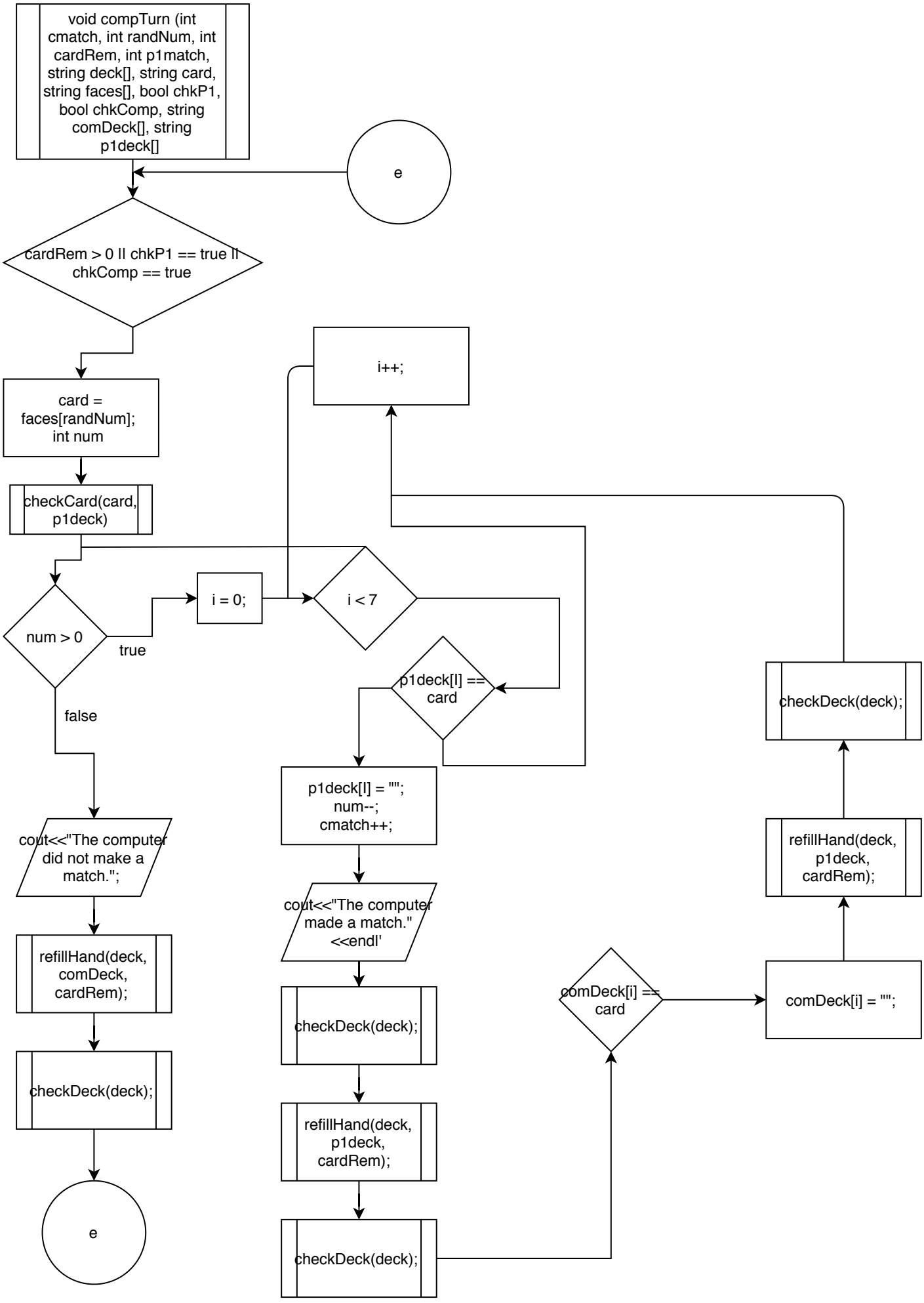

void dealDeck(string deck[], string p1deck[], string comDeck[])
int i=0;
i++;
I<7
true
p1deck[I] = deck[I];
comDeck[I] = deck[I+7];
deck[I] = "";
deck[I+7] = "";
void remCardsP1(string deck[], string p1deck[])
int i=0;
i++;
I<13
true
false
deck[I] == p1deck[I];
true
deck[I] = "";
void showDeck(string deck[])
int i=0;
i++;
I<7
true
cout<<deck[i]<<" ";
cout<<endl;
void remCardComp1(string deck[], string comDeck[])
int i=0;
i++;
I<13
true
false
deck[I] == comDeck[I];
true
deck[I] = "";


int checkMatches(string array[])
int i;
int b = 0;
i++;
i<6
false
return b;
true
int j = i;
b++;
array[l] = "";
array[j] = "";
j<6;
true
j != i
true
array[l] == array[j] && array[l] != ""
true
j++;


void refillHand(string deck[], string array[], int cardRem)
int i=cardRem;
j = 0;
j++;
j<7
true
false
array[j] == ""
true
array[j] = deck[l];
deck[l] = "";
cardRem--;
int checkCard(string card, string deck[])
int num = 0;
int i = 0;
i++
false
num++
deck[i] == card
true
l<7
true
return num;
int checkDeck(string deck[])
int num = 0;
int i = 0;
i++
i<52
true
false
num++
deck[i] == "";
true
return (52-num);


void p1Turn (int cardRem, int p1match, string deck[], string card, string faces[], bool chkP1, bool chkComp, string comDeck[], string p1Deck[])
d
cardRem > 0 || chkP1 == true || chkComp == true;
bool chkCard = false;
chkCard == false;
true
cout<<"Player 1 deck: ";
false
c
showDeck(p1deck);
checkMatches(p1deck);
checkDeck(deck);
refillHand(deck, p1deck, cardRem);
cout<<"Which card would you like to ask the computer for: \n";
cout<<"Spell out the number or face card with standard capitalization.\n";
cin>>card;
cout<<"Invalid input" <<endl;
chkCard == false
chkCard = true;
int I = 0;
I < 13
true
false
card == faces[I]
i++;


c
int num;
bool check;
checkCard(card, comDeck);
num > 0
true
int i = 0;
false
num == 0
d
cout<<"Go Fish!" <<endl;
checkDeck(deck);
refillHand(deck, p1deck, cardRem)
i++;
i < 7
false
true
comDeck[I] == card
false
check
false
true
comDeck[I] = "";
num--;
p1match++;
check = true;
checkDeck(deck);
refillHand(deck, comDeck, cardRem);
check = true;
checkDeck(deck);
true
p1deck[I] == card
false
true
p1deck[i] = "";
refillHand(deck, comDeck, cardRem);
checkDeck(deck);


void compTurn (int cmatch, int randNum, int cardRem, int p1match, string deck[], string card, string faces[], bool chkP1, bool chkComp, string comDeck[], string p1deck[]
e
cardRem > 0 || chkP1 == true || chkComp == true
card = faces[randNum]; int num
checkCard(card, p1deck)
num > 0
true
i = 0;
i < 7
i++;
p1deck[I] == card
p1deck[I] = ""; num--; cmatch++;
cout<<"The computer made a match." <<endl'
checkDeck(deck);
refillHand(deck, p1deck, cardRem);
checkDeck(deck);
comDeck[i] == card
comDeck[i] = "";
refillHand(deck, p1deck, cardRem);
checkDeck(deck);
false
cout<<"The computer did not make a match.";
refillHand(deck, comDeck, cardRem);
checkDeck(deck);
e