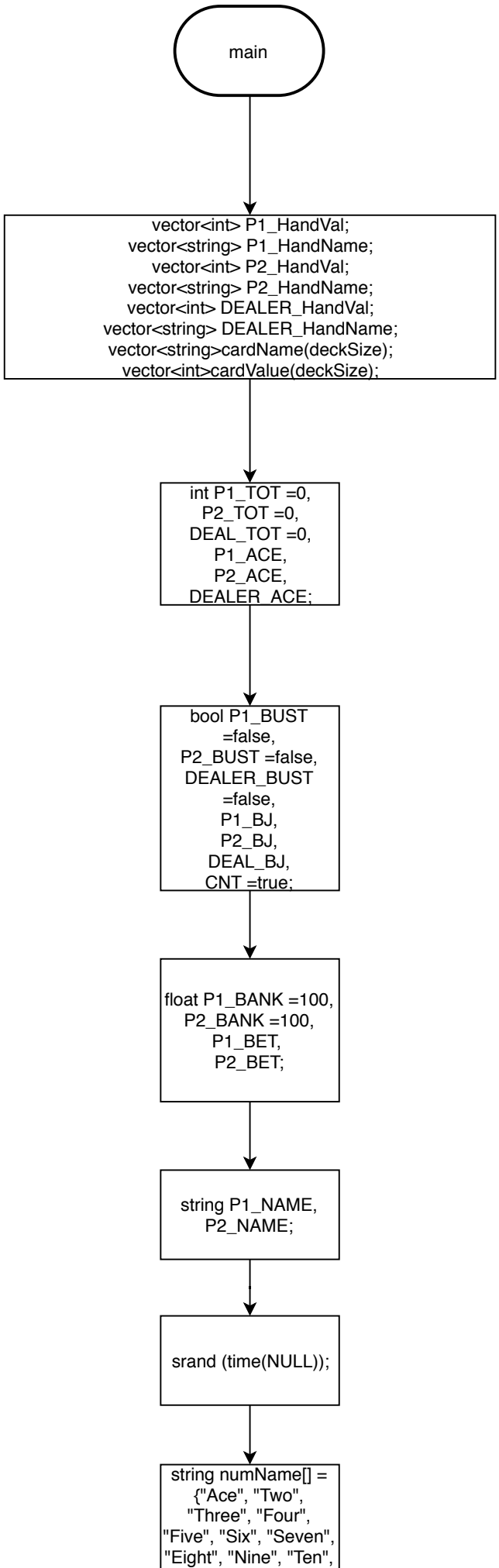
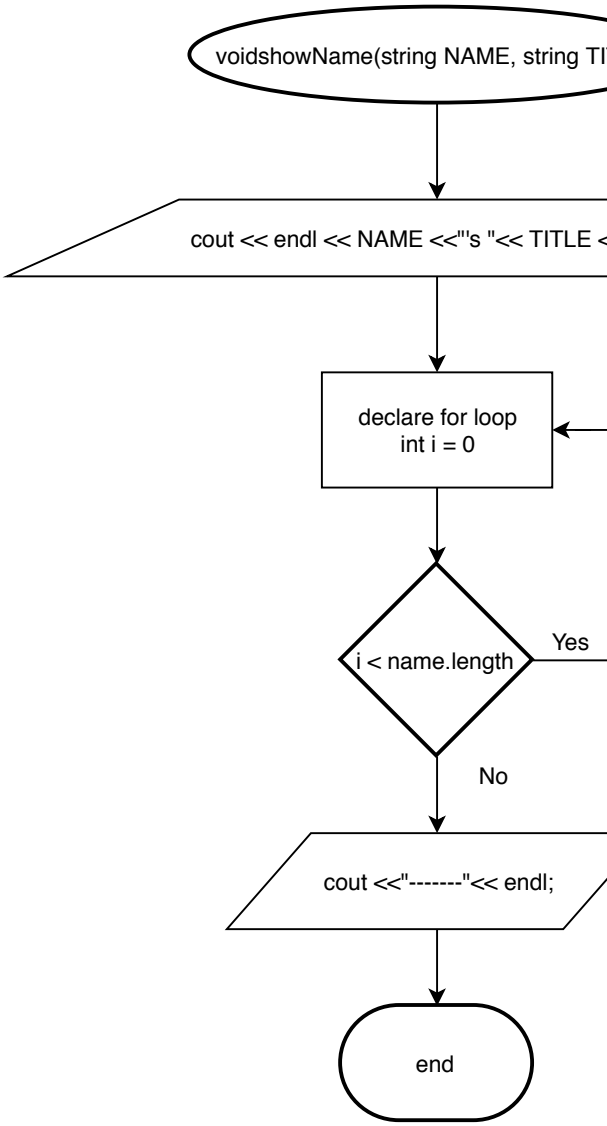
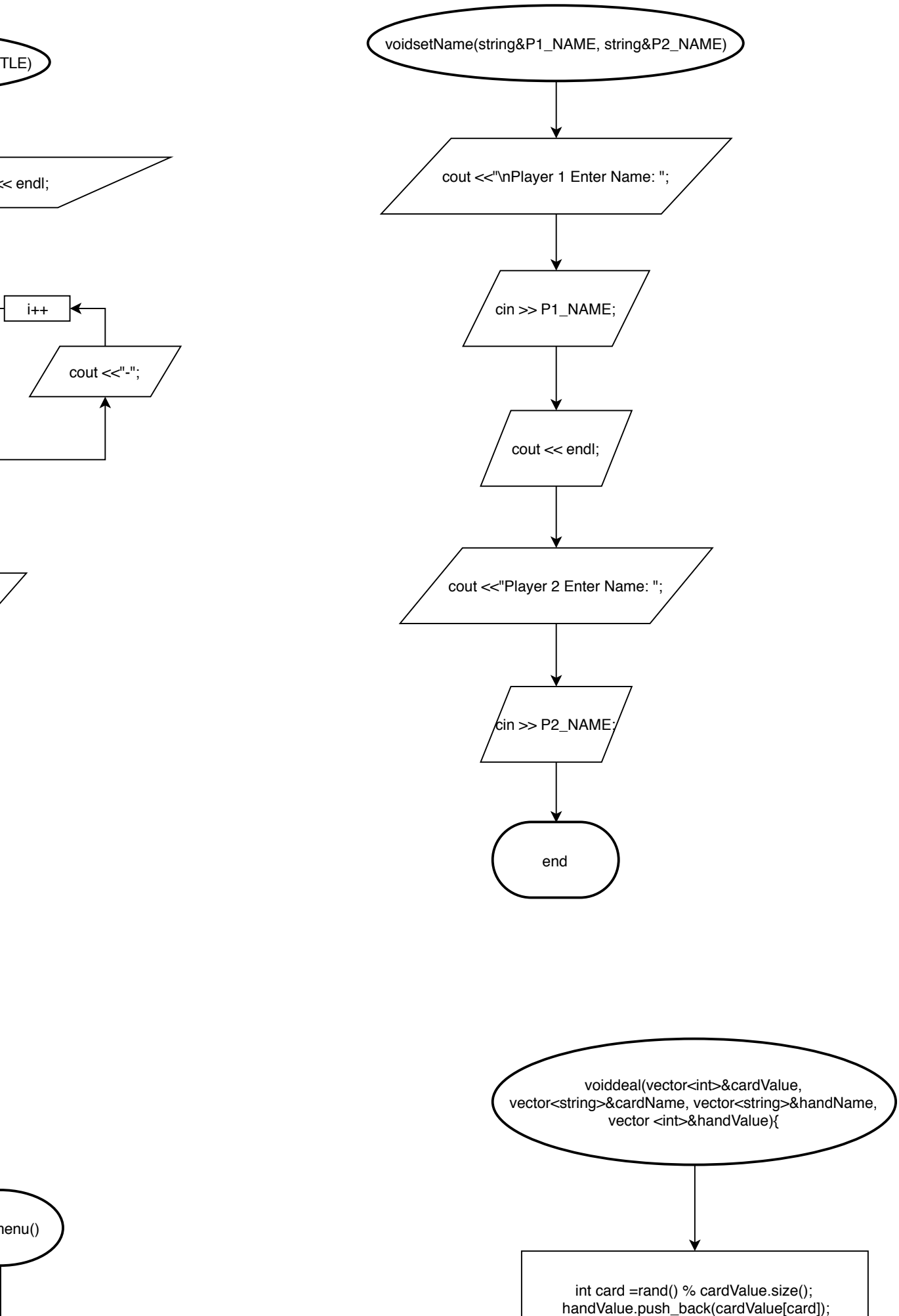
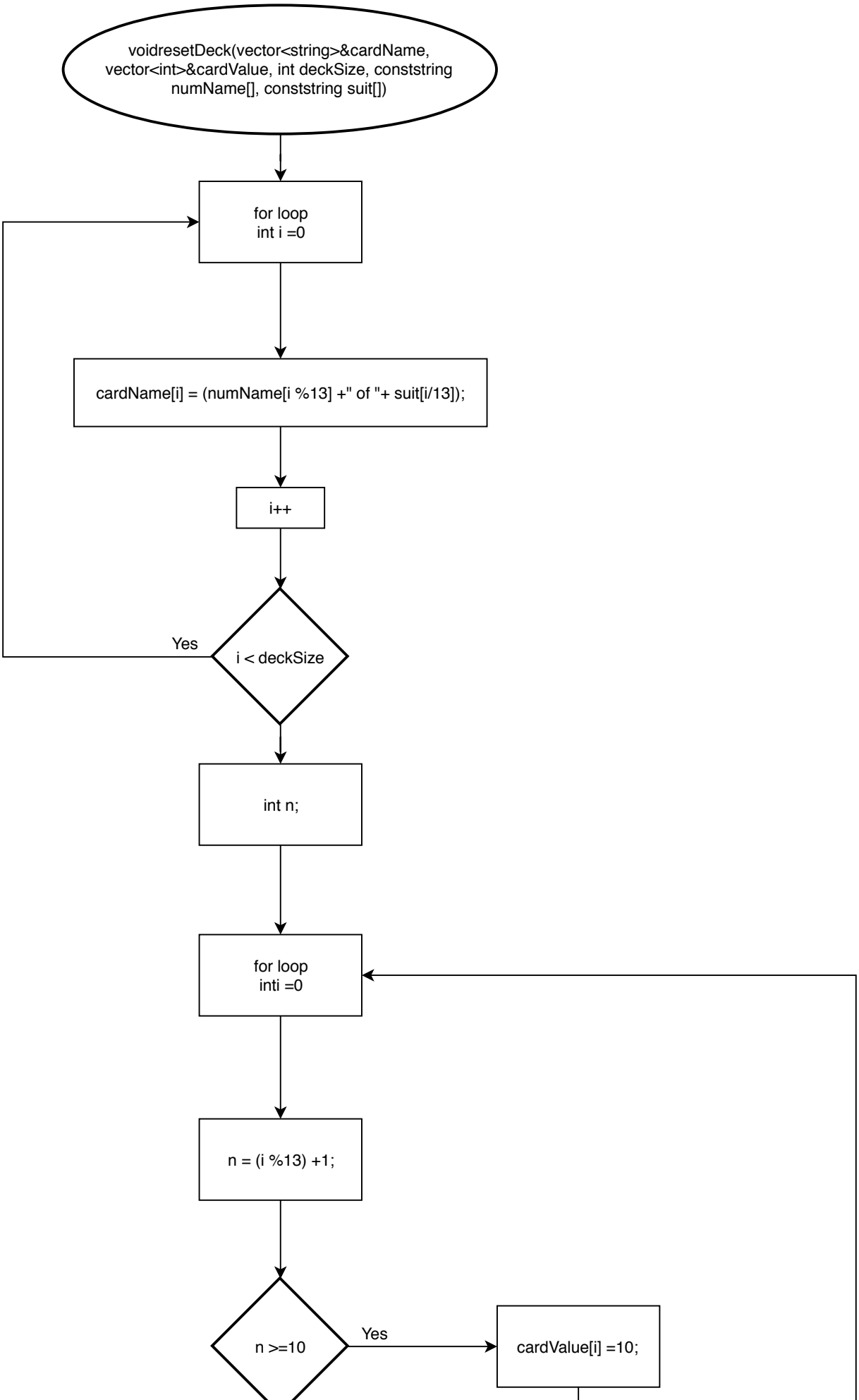
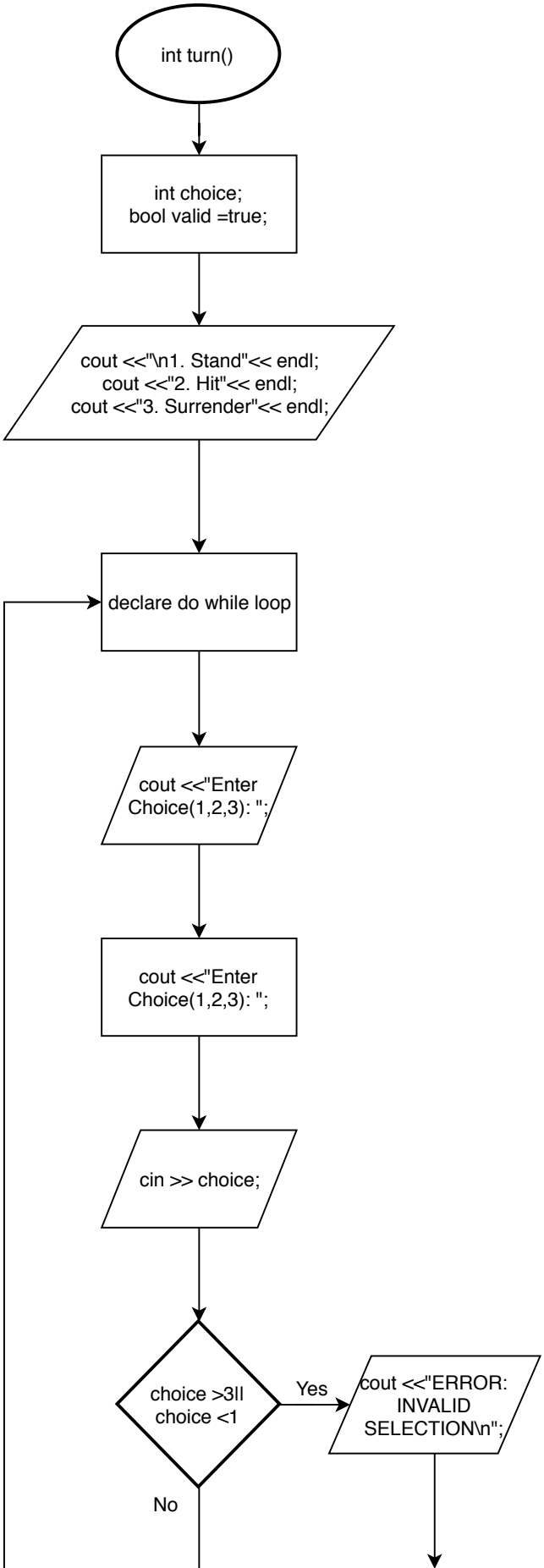

main
vector<int> P1_HandVal;
vector<string> P1_HandName;
vector<int> P2_HandVal;
vector<string> P2_HandName;
vector<int> DEALER_HandVal;
vector<string> DEALER_HandName;
vector<string>cardName(deckSize);
vector<int>cardValue(deckSize);
int P1_TOT =0,
P2_TOT =0,
DEAL_TOT =0,
P1_ACE,
P2_ACE,
DEALER_ACE;
bool P1_BUST
=false,
P2_BUST =false,
DEALER_BUST
=false,
P1_BJ,
P2_BJ,
DEAL_BJ,
CNT =true;
float P1_BANK =100,
P2_BANK =100,
P1_BET,
P2_BET;
string P1_NAME,
P2_NAME;
srand (time(NULL));
string numName[] =
{"Ace", "Two",
"Three", "Four",
"Five", "Six", "Seven",
"Eight", "Nine", "Ten",

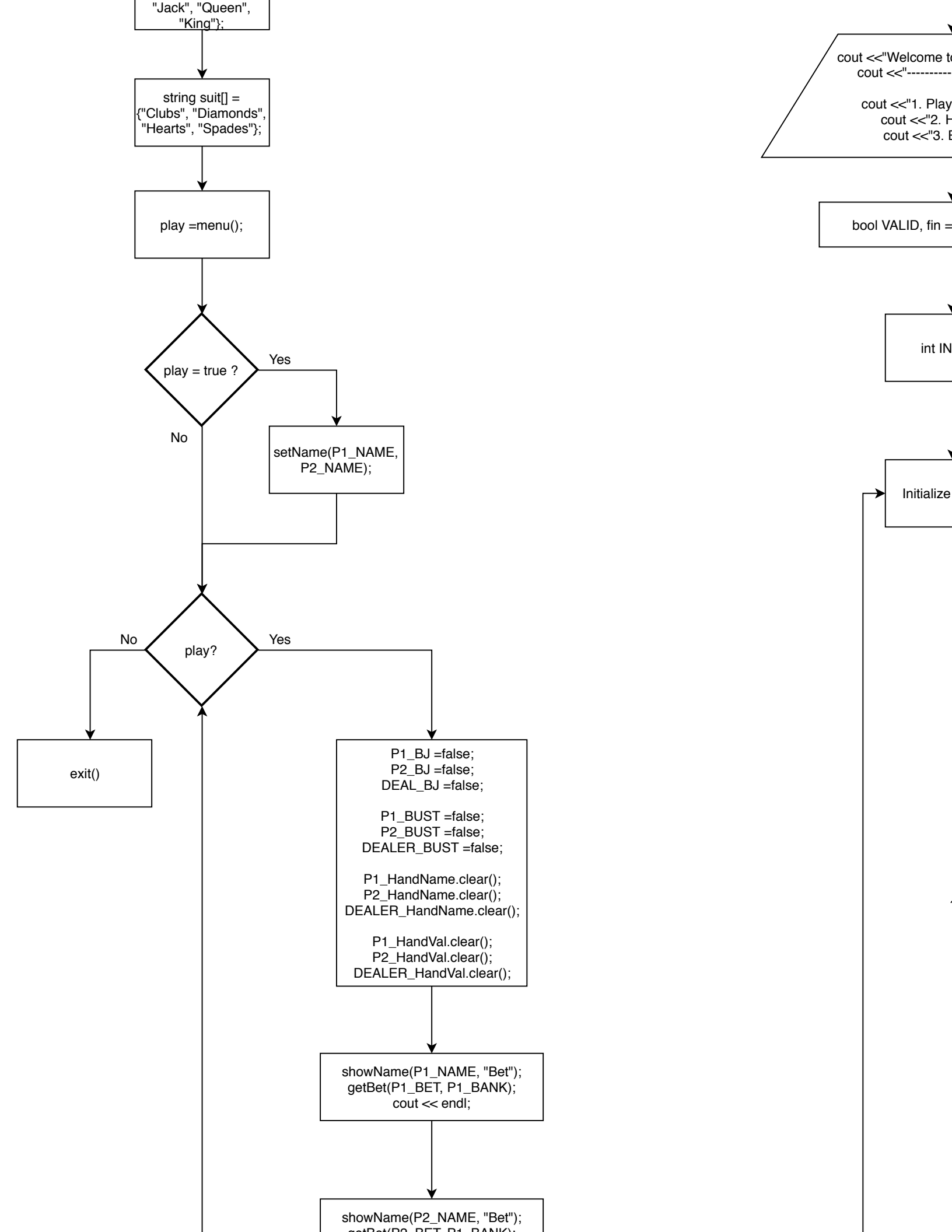

voidshowName(string NAME, string TI
cout << endl << NAME <<"'s "<< TITLE <
declare for loop
int i = 0
i < name.length
Yes
No
cout <<"-------"<< endl;
end

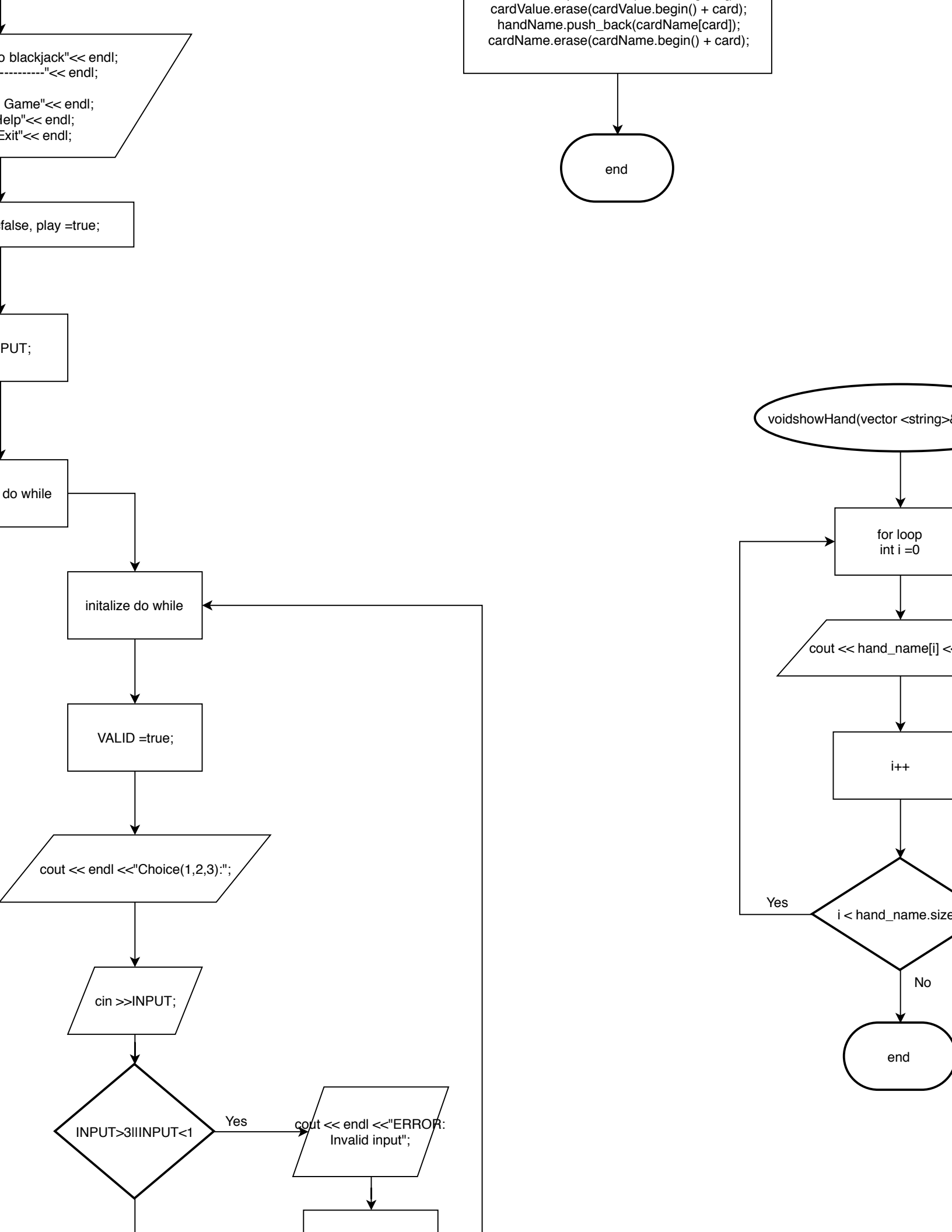
bool m

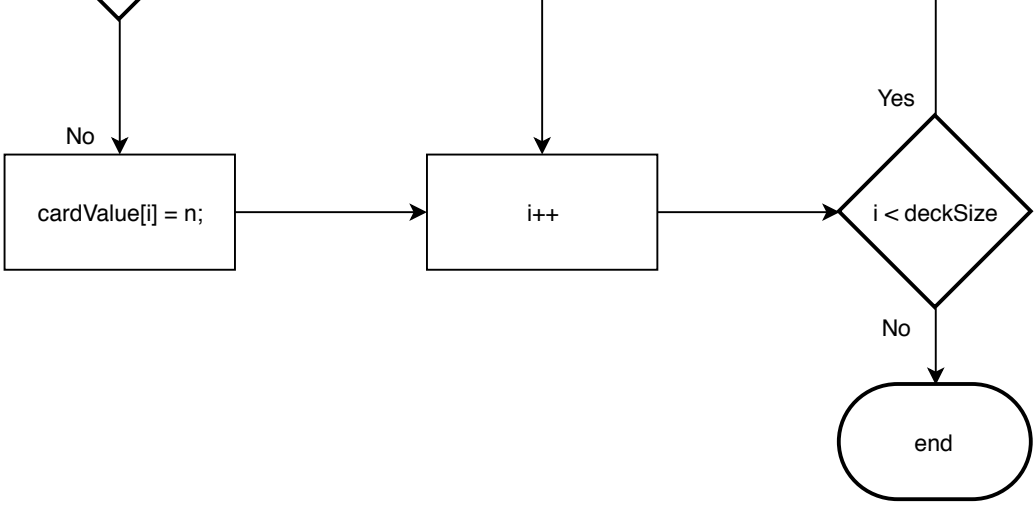

TLE)
<< endl;
i++
cout <<"-";
voidsetName(string&P1_NAME, string&P2_NAME)
cout <<"\nPlayer 1 Enter Name: ";
cin >> P1_NAME;
cout << endl;
cout <<"Player 2 Enter Name: ";
cin >> P2_NAME;
end
menu()
voiddeal(vector<int>&cardValue,
vector<string>&cardName, vector<string>&handName,
vector <int>&handValue){
int card =rand() % cardValue.size();
handValue.push_back(cardValue[card]);

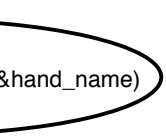

voidresetDeck(vector<string>&cardName, vector<int>&cardValue, int deckSize, conststring numName[], conststring suit[])
for loop int i =0
cardName[i] = (numName[i %13] +" of "+ suit[i/13]);
i++
Yes
i < deckSize
int n;
for loop inti =0
n = (i %13) +1;
n >=10
Yes
cardValue[i] =10;

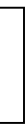

int turn()
int choice;
bool valid =true;
cout <<"\n1. Stand"<< endl;
cout <<"2. Hit"<< endl;
cout <<"3. Surrender"<< endl;
declare do while loop
cout <<"Enter Choice(1,2,3): ";
cout <<"Enter Choice(1,2,3): ";
cin >> choice;
choice >3|| choice <1
Yes
cout <<"ERROR: INVALID SELECTION\n";
No

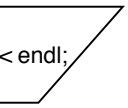

"Jack", "Queen", "King"};
string suit[] = {"Clubs", "Diamonds", "Hearts", "Spades"};
play =menu();
play = true ?
Yes
No
setName(P1_NAME, P2_NAME);
play?
No
Yes
exit()
P1_BJ =false;
P2_BJ =false;
DEAL_BJ =false;
P1_BUST =false;
P2_BUST =false;
DEALER_BUST =false;
P1_HandName.clear();
P2_HandName.clear();
DEALER_HandName.clear();
P1_HandVal.clear();
P2_HandVal.clear();
DEALER_HandVal.clear();
showName(P1_NAME, "Bet");
getBet(P1_BET, P1_BANK);
cout << endl;
showName(P2_NAME, "Bet");
cout <<"Welcome t
cout <<"----------
cout <<"1. Play
cout <<"2. H
cout <<"3. E
bool VALID, fin =
int IN
Initialize

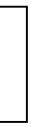

cardValue.erase(cardValue.begin() + card);
handName.push_back(cardName[card]);
cardName.erase(cardName.begin() + card);
end
o blackjack"<< endl;
----------"<< endl;
Game"<< endl;
Help"<< endl;
Exit"<< endl;
false, play =true;
PUT;
do while
initalize do while
VALID =true;
cout << endl <<"Choice(1,2,3):";
cin >>INPUT;
INPUT>3||INPUT<1
Yes
cout << endl <<"ERROR:
Invalid input";
voidshowHand(vector <string>&
for loop
int i =0
cout << hand_name[i] <<
i++
Yes
i < hand_name.size
No
end


No
cardValue[i] = n;
i++
Yes
i < deckSize
No
end

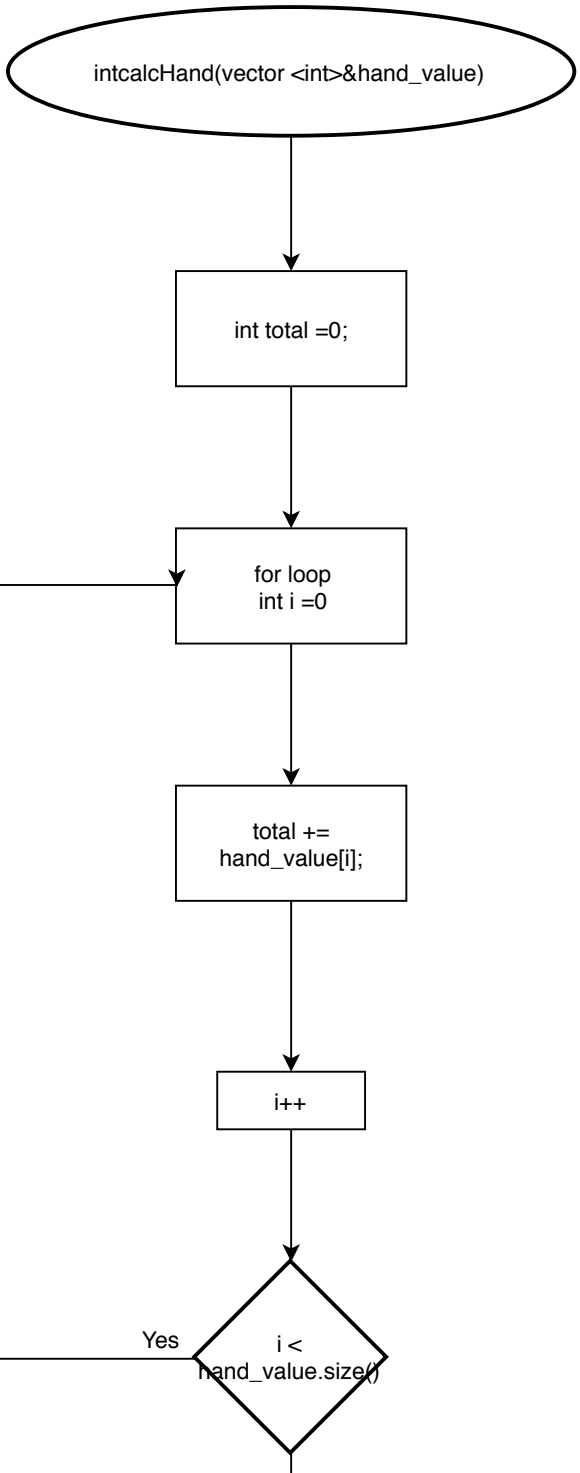

&hand_name)

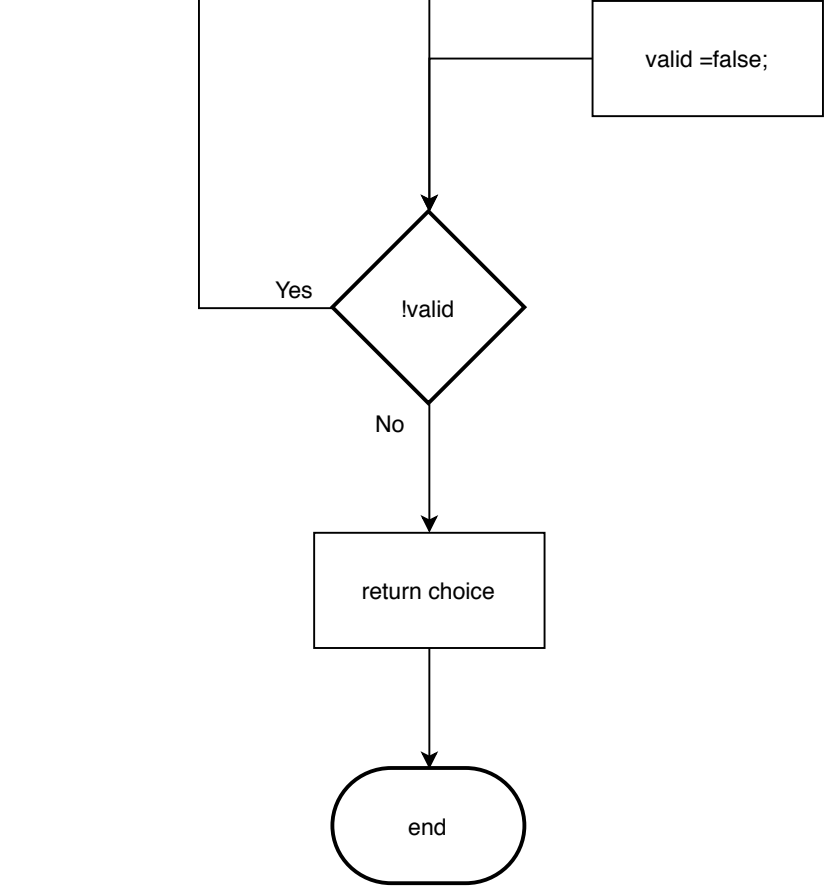


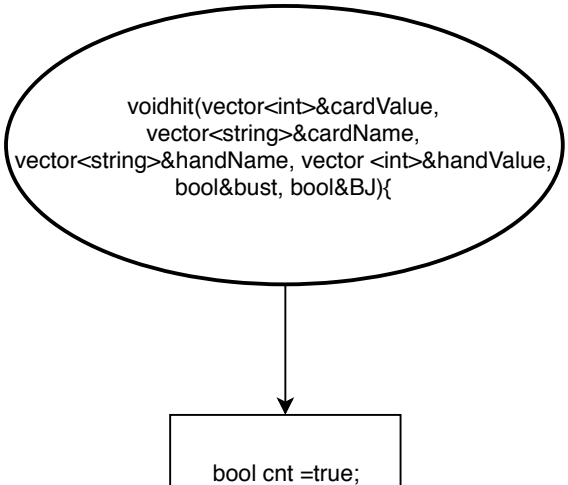

< endl;

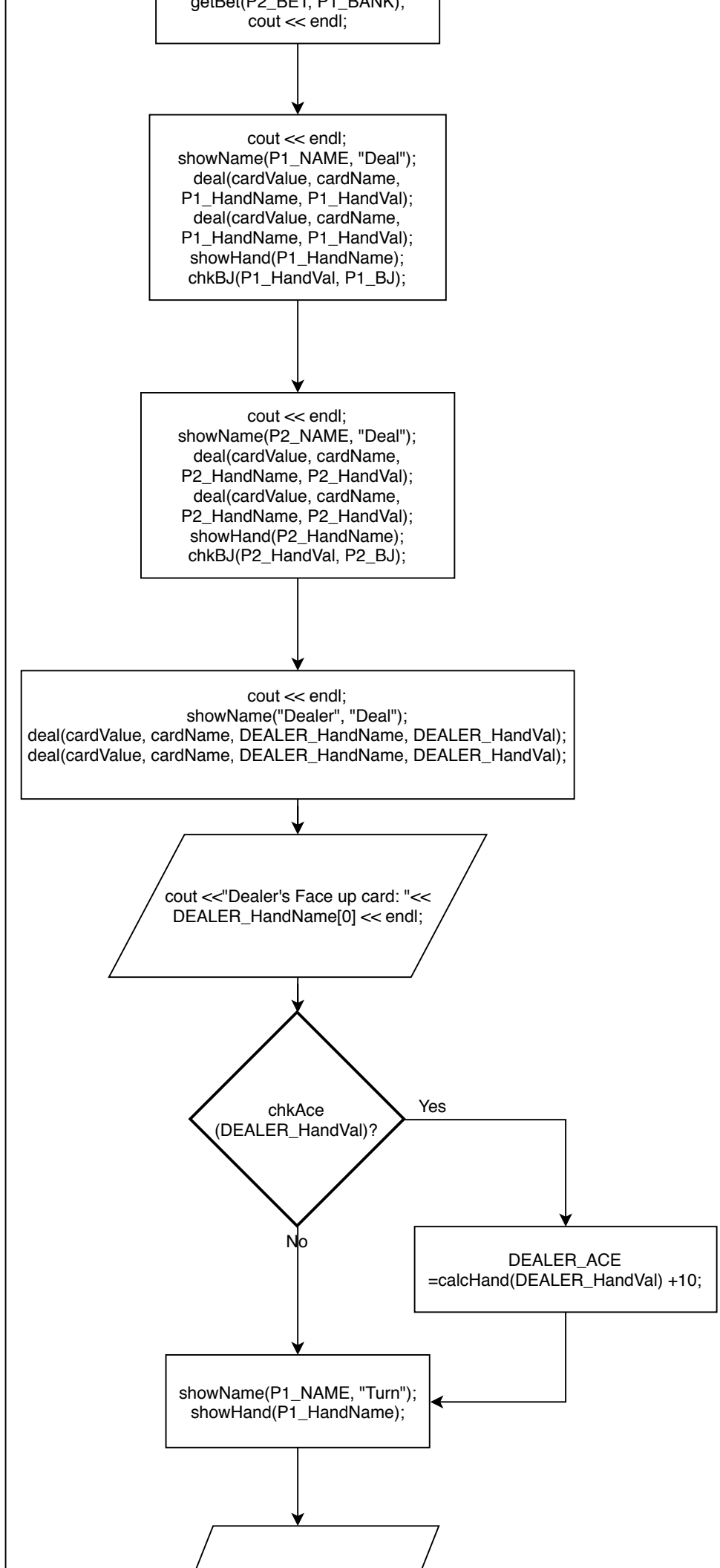


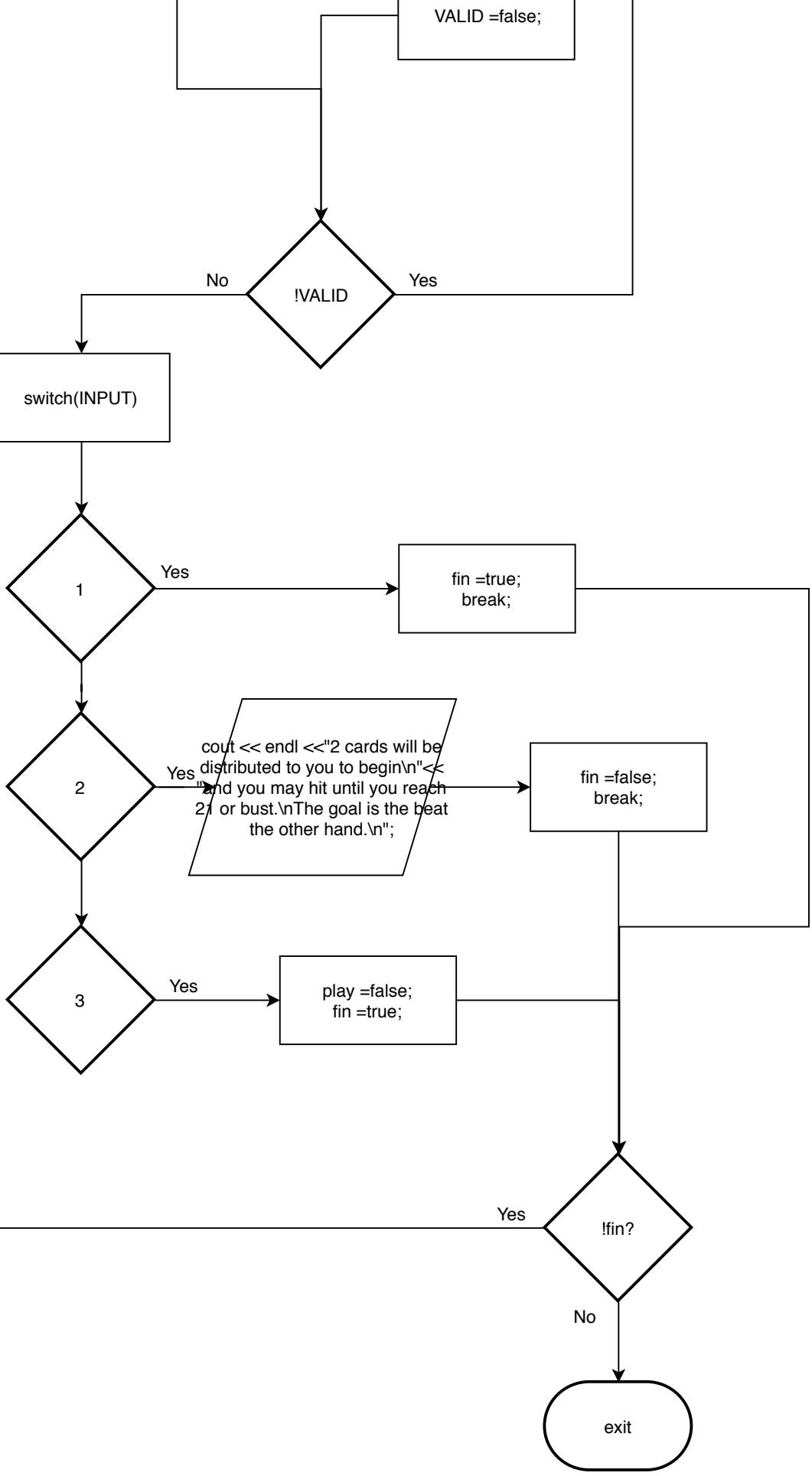
e()

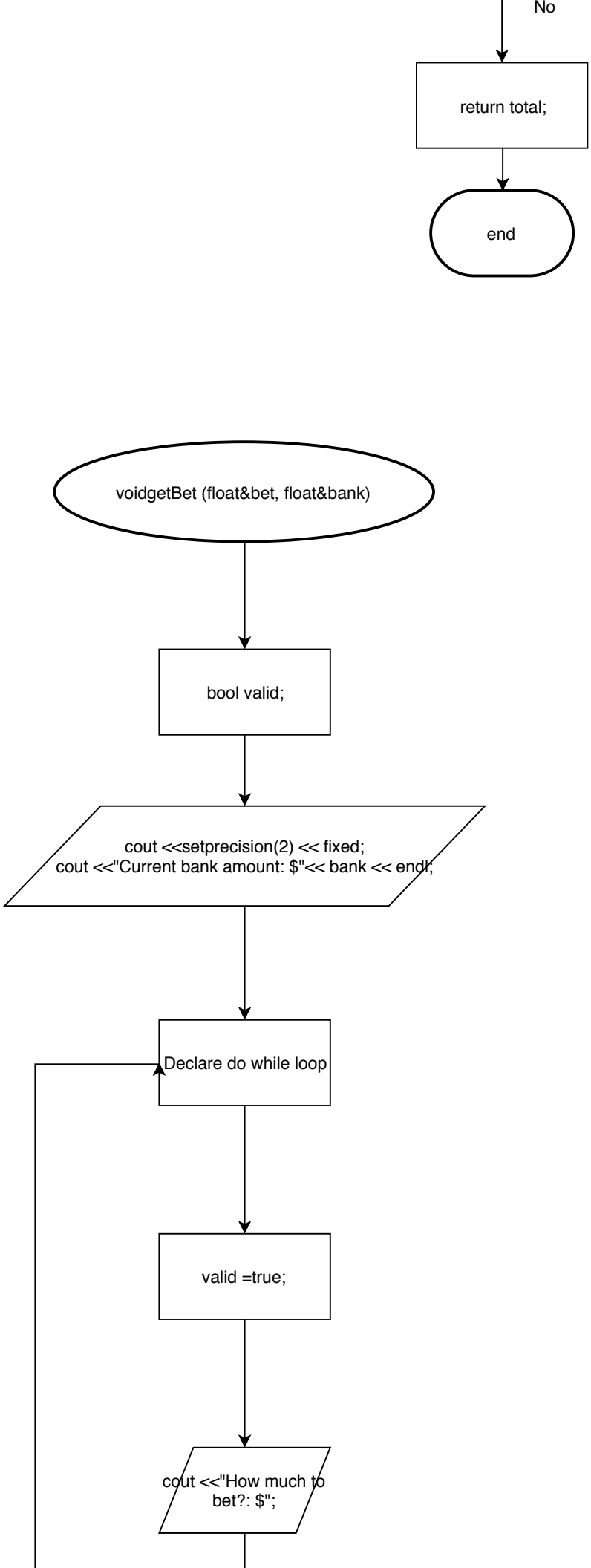

intcalcHand(vector <int>&hand_value)
int total =0;
for loop
int i =0
total +=
hand_value[i];
i++
Yes
i <
hand_value.size()

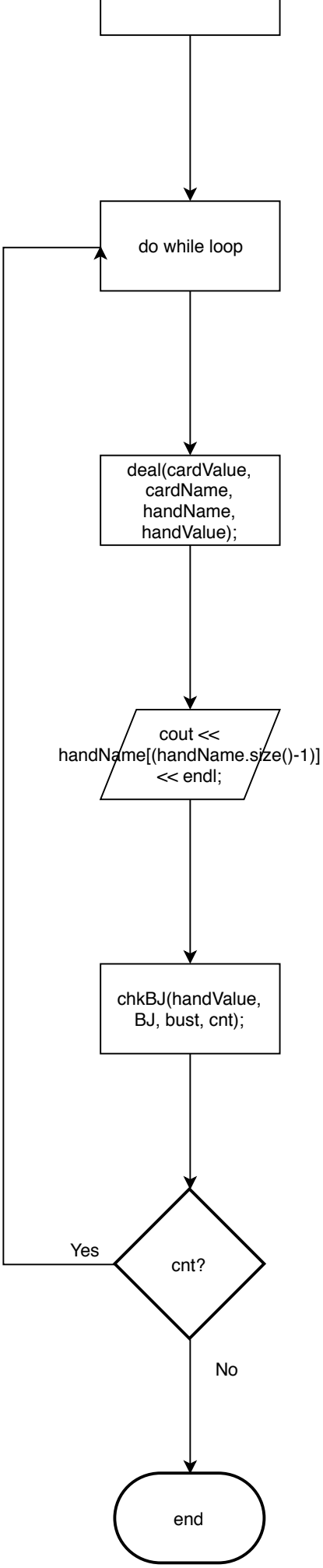

valid =false;
Yes
!valid
No
return choice
end

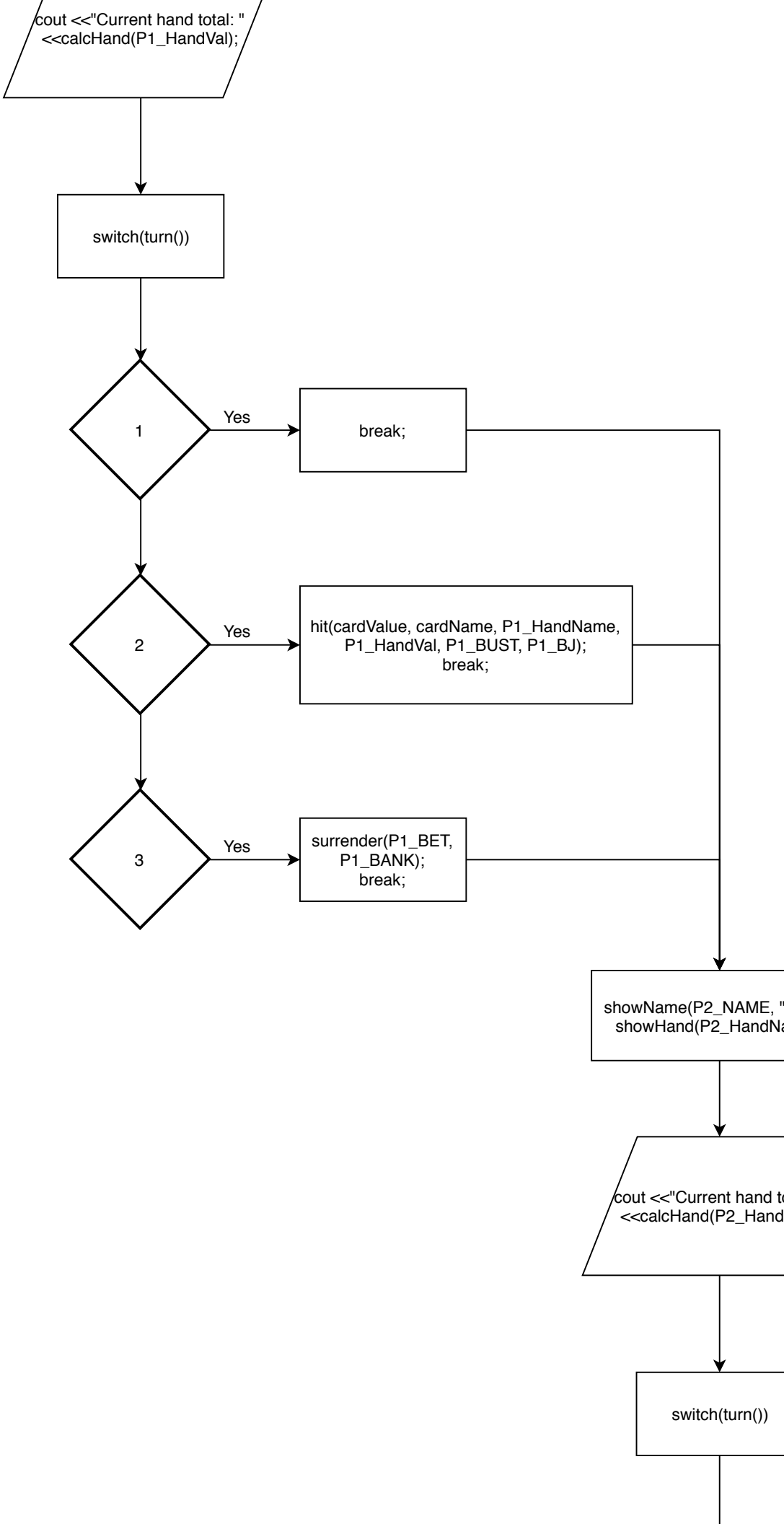

voidhit(vector<int>&cardValue,
vector<string>&cardName,
vector<string>&handName, vector <int>&handValue,
bool&bust, bool&BJ){
bool cnt =true;

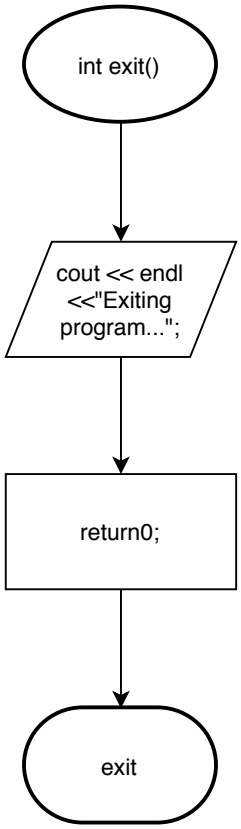

getBet(P2_BET, P1_BANK);
cout << endl;
cout << endl;
showName(P1_NAME, "Deal");
deal(cardValue, cardName,
P1_HandName, P1_HandVal);
deal(cardValue, cardName,
P1_HandName, P1_HandVal);
showHand(P1_HandName);
chkBJ(P1_HandVal, P1_BJ);
cout << endl;
showName(P2_NAME, "Deal");
deal(cardValue, cardName,
P2_HandName, P2_HandVal);
deal(cardValue, cardName,
P2_HandName, P2_HandVal);
showHand(P2_HandName);
chkBJ(P2_HandVal, P2_BJ);
cout << endl;
showName("Dealer", "Deal");
deal(cardValue, cardName, DEALER_HandName, DEALER_HandVal);
deal(cardValue, cardName, DEALER_HandName, DEALER_HandVal);
cout <<"Dealer's Face up card: "<<
DEALER_HandName[0] << endl;
chkAce
(DEALER_HandVal)?
Yes
No
DEALER_ACE
=calcHand(DEALER_HandVal) +10;
showName(P1_NAME, "Turn");
showHand(P1_HandName);

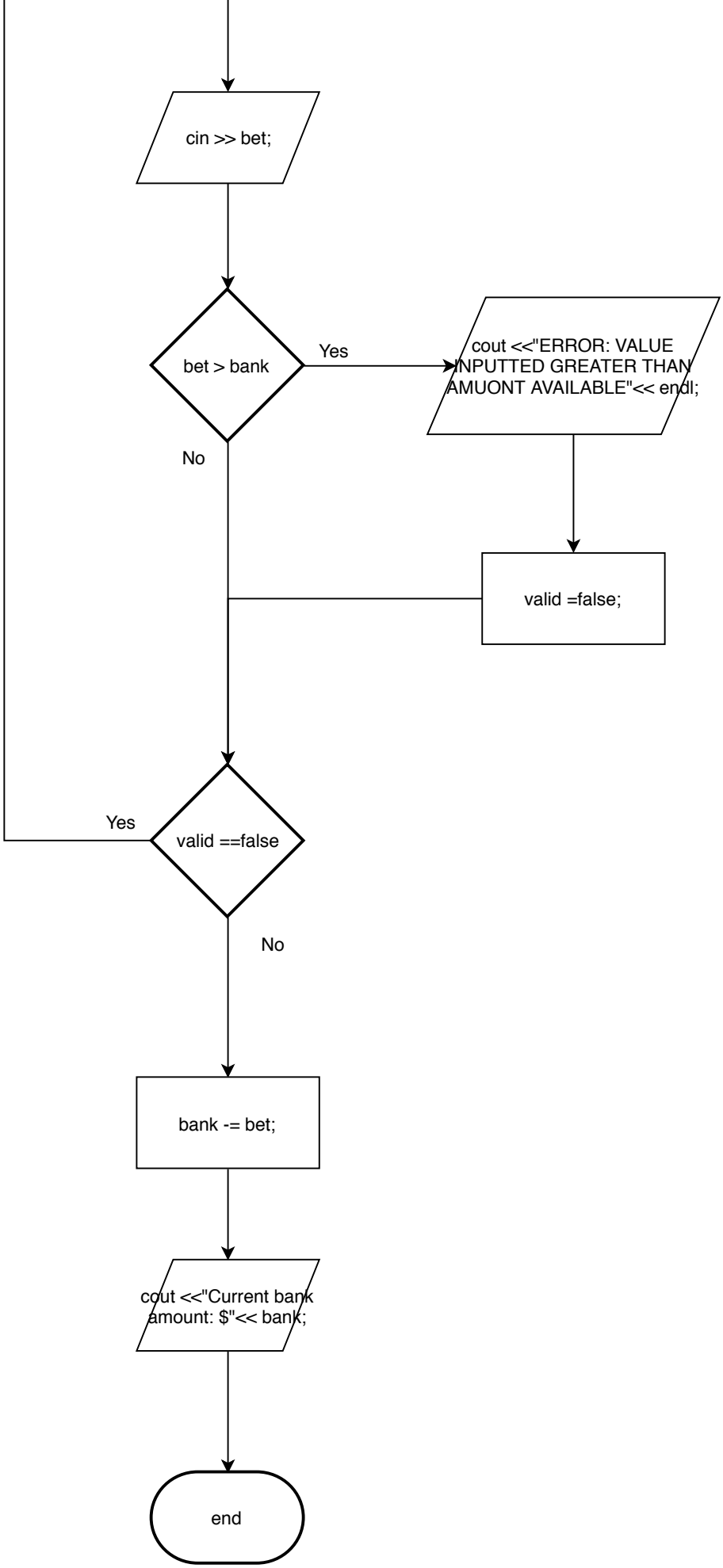

VALID =false;
!VALID
No
Yes
switch(INPUT)
1
Yes
fin =true;
break;
2
Yes
cout << endl <<"2 cards will be distributed to you to begin\n"<< "and you may hit until you reach 21 or bust.\nThe goal is the beat the other hand.\n";
fin =false;
break;
3
Yes
play =false;
fin =true;
!fin?
Yes
No
exit

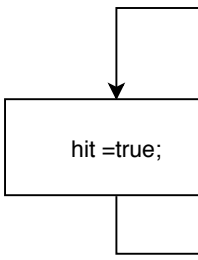

No
return total;
end
voidgetBet (float&bet, float&bank)
bool valid;
cout <<setprecision(2) << fixed;
cout <<"Current bank amount: $"<< bank << endl;
Declare do while loop
valid =true;
cout <<"How much to bet?: $";

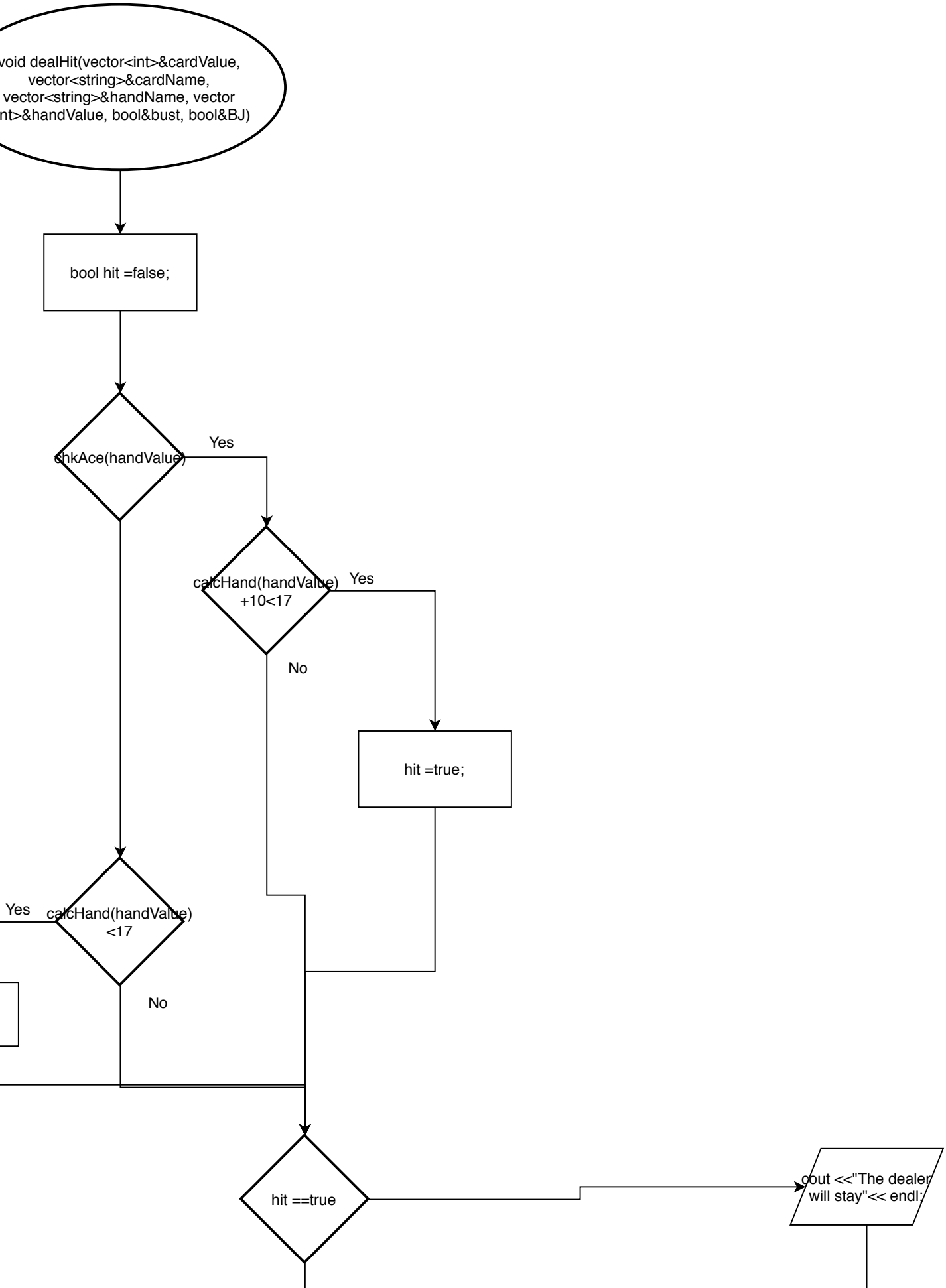

do while loop
deal(cardValue, cardName, handName, handValue);
cout << handName[(handName.size()-1)] << endl;
chkBJ(handValue, BJ, bust, cnt);
cnt?
Yes
No
end

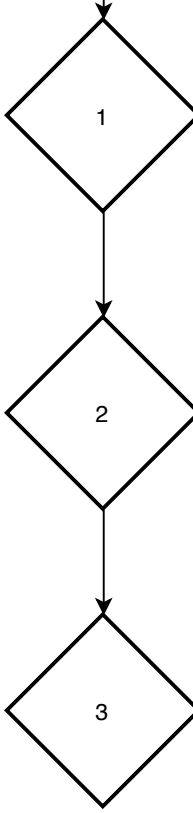

cout <<"Current hand total: "
<<calcHand(P1_HandVal);
switch(turn())
1
Yes
break;
2
Yes
hit(cardValue, cardName, P1_HandName,
P1_HandVal, P1_BUST, P1_BJ);
break;
3
Yes
surrender(P1_BET,
P1_BANK);
break;
showName(P2_NAME, "
showHand(P2_HandNa
cout <<"Current hand t
<<calcHand(P2_Hand
switch(turn())

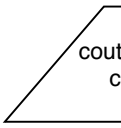

int exit()
cout << endl
<<"Exiting
program...";
return0;
exit

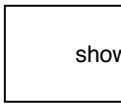

cin >> bet;
bet > bank
Yes
cout <<"ERROR: VALUE INPUTTED GREATER THAN AMUONT AVAILABLE"<< endl;
No
valid =false;
Yes
valid ==false
No
bank -= bet;
cout <<"Current bank amount: $"<< bank;
end

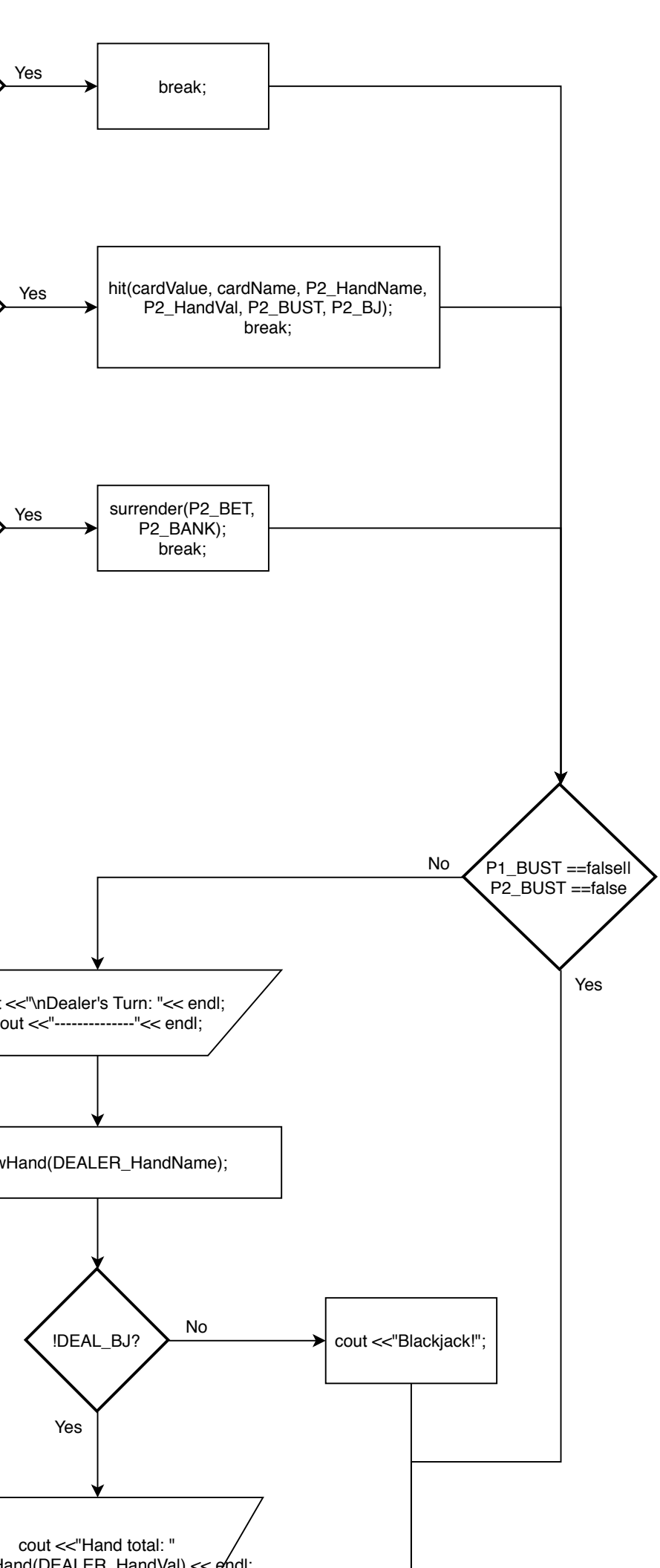

hit =true;

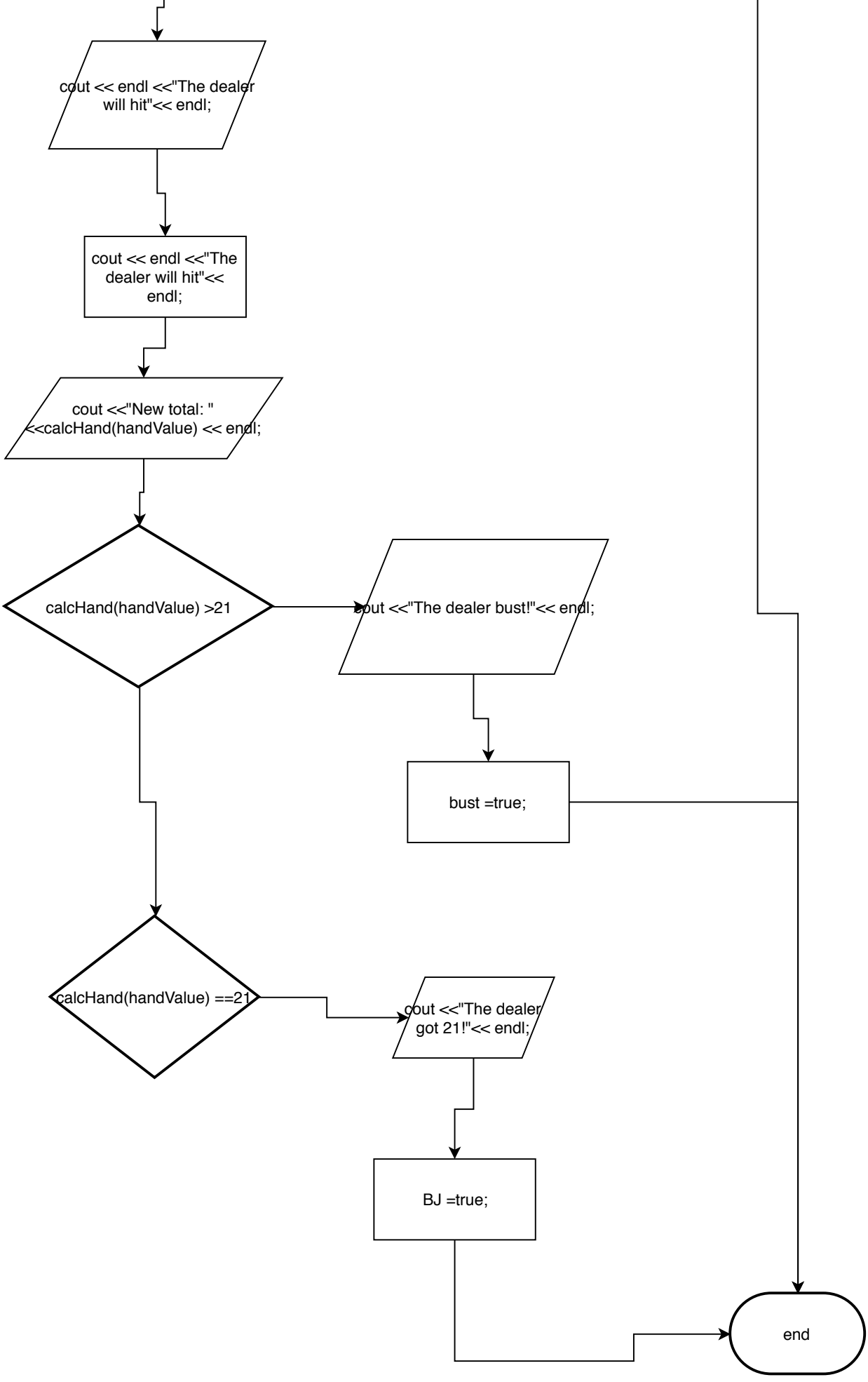

void dealHit(vector<int>&cardValue,
vector<string>&cardName,
vector<string>&handName, vector
nt>&handValue, bool&bust, bool&BJ)
bool hit =false;
chkAce(handValue)
Yes
calcHand(handValue)
+10<17
Yes
No
hit =true;
Yes
calcHand(handValue)
<17
No
hit ==true
cout <<"The dealer
will stay"<< endl;


1
2
3

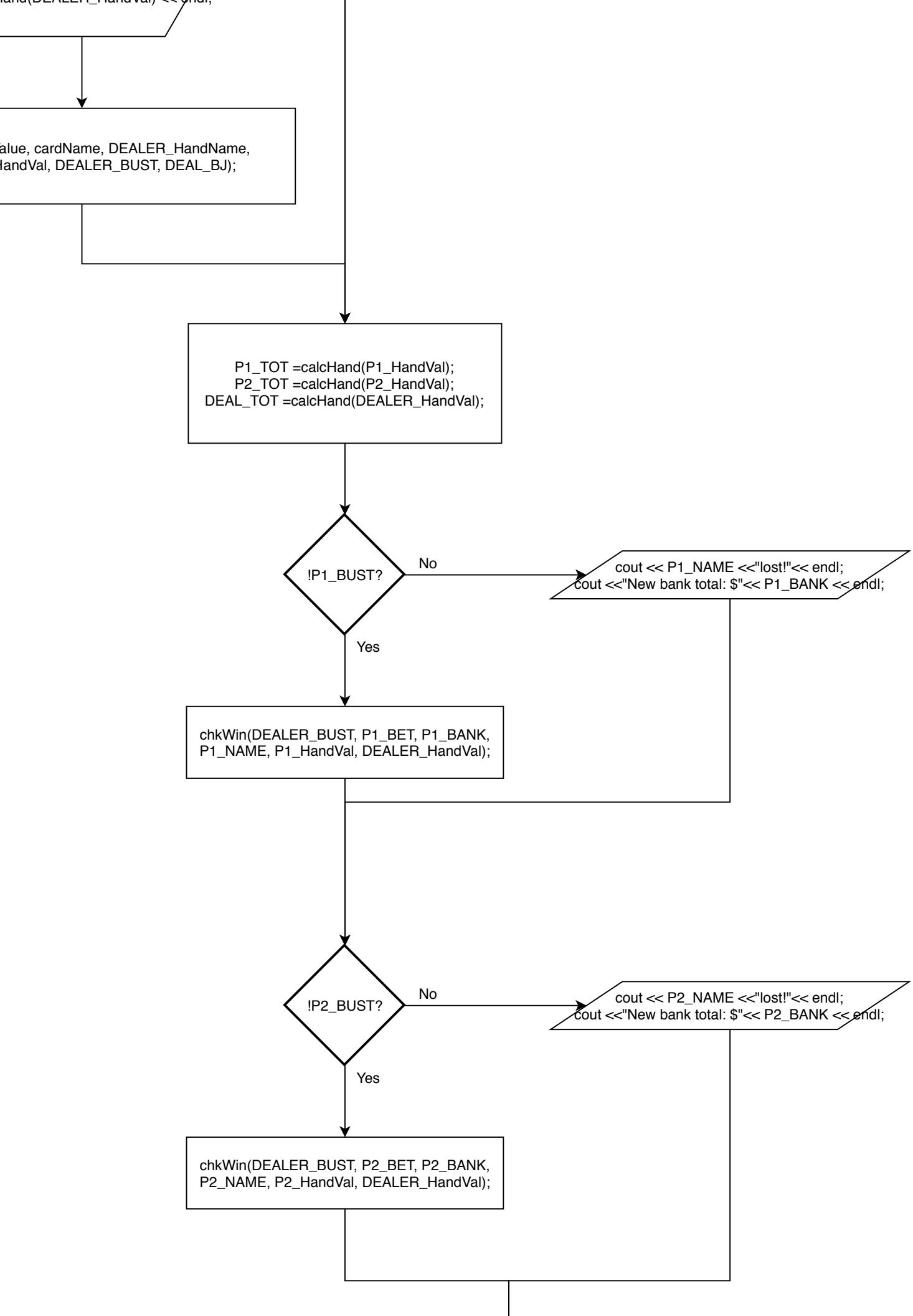

cout
c

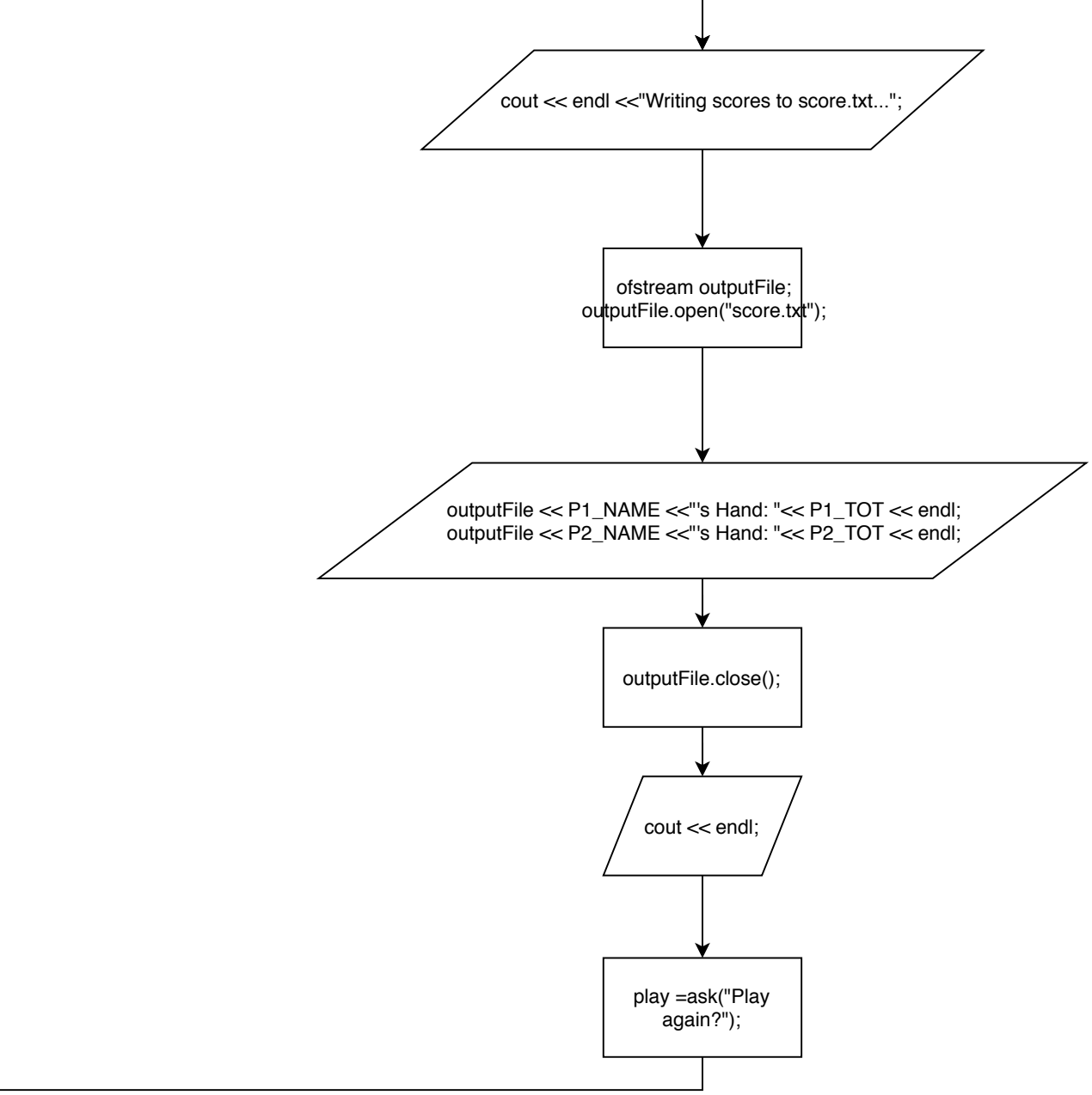

show


Yes
break;
Yes
hit(cardValue, cardName, P2_HandName,
P2_HandVal, P2_BUST, P2_BJ);
break;
Yes
surrender(P2_BET,
P2_BANK);
break;
No
P1_BUST ==falseII
P2_BUST ==false
Yes
<<"\nDealer's Turn: "<< endl;
out <<"----------"<< endl;
wHand(DEALER_HandName);
!DEAL_BJ?
No
cout <<"Blackjack!";
Yes
cout <<"Hand total: "


cout << endl <<"The dealer will hit"<< endl;
cout << endl <<"The dealer will hit"<< endl;
cout <<"New total: " <<calcHand(handValue) << endl;
calcHand(handValue) >21
cout <<"The dealer bust!"<< endl;
bust =true;
calcHand(handValue) ==21
cout <<"The dealer got 21!"<< endl;
BJ =true;
end

dealHit(cardV
DEALER_H


alue, cardName, DEALER_HandName,
HandVal, DEALER_BUST, DEAL_BJ);
P1_TOT =calcHand(P1_HandVal);
P2_TOT =calcHand(P2_HandVal);
DEAL_TOT =calcHand(DEALER_HandVal);
!P1_BUST?
No
cout << P1_NAME <<"lost!"<< endl;
cout <<"New bank total: $"<< P1_BANK << endl;
Yes
chkWin(DEALER_BUST, P1_BET, P1_BANK,
P1_NAME, P1_HandVal, DEALER_HandVal);
!P2_BUST?
No
cout << P2_NAME <<"lost!"<< endl;
cout <<"New bank total: $"<< P2_BANK << endl;
Yes
chkWin(DEALER_BUST, P2_BET, P2_BANK,
P2_NAME, P2_HandVal, DEALER_HandVal);


cout << endl <<"Writing scores to score.txt...";
ofstream outputFile;
outputFile.open("score.txt");
outputFile << P1_NAME <<"'s Hand: "<< P1_TOT << endl;
outputFile << P2_NAME <<"'s Hand: "<< P2_TOT << endl;
outputFile.close();
cout << endl;
play =ask("Play again?");