**C Programming**

13016235

**Project: Blackjack Let’s get rich**

Name: Chanyanuch Likitpanjamanon

ID: 61090004

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <ctype.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <stdbool.h>

#include <dos.h>

#define SUIT 13

#define CARDS 52

static int win = 0;

struct Player

{

char name[100];

int money;

int Win;

int point;

int point2;

char cardPlayer[CARDS][2];

char cardPlayer2[CARDS][2];

};

struct Dealer

{

char cardDealer[CARDS][2];

int point;

};

int displayFile(char \*filename)

{

FILE \*inFile;

char a;

inFile = fopen(filename, "r");

if (inFile == NULL)

{

return 0;

}

while (fscanf(inFile, "%c", &a) != EOF)

{

printf("%c", a);

}

fclose(inFile);

return 1;

}

int updateFile(char name1[50] , int winF , int moneyF , struct Player player)

{

FILE \*fp, \*fp1;

int res, f = 0;

fp = fopen("userData.txt", "r");

fp1 = fopen("temp.txt", "a");

while (fscanf(fp, "%s %d %d", name1, &winF, &moneyF) != EOF)

{

res = strcmp(player.name, name1);

if (res == 0)

{

f = 1;

fprintf(fp1, "%s %d %d\n", name1, player.Win, player.money);

}

else

{

fprintf(fp1, "%s %d %d\n", name1, winF, moneyF);

}

}

if (f == 0)

{

printf("Name not found");

}

fclose(fp);

fclose(fp1);

fp = fopen("userData.txt", "w");

fclose(fp);

fp = fopen("userData.txt", "a");

fp1 = fopen("temp.txt", "r");

while (fscanf(fp1, "%s %d %d\n", name1, &winF, &moneyF) != EOF)

{

fprintf(fp, "%s %d %d\n", name1, winF, moneyF);

}

fclose(fp);

fclose(fp1);

fp = fopen("temp.txt", "w");

fclose(fp);

}

int writeFile(char \*name,int count,int credit)

{

FILE \*outFile;

char filename2[1000] = "userData.txt";

outFile = fopen(filename2, "a");

if (outFile != NULL)

{

fprintf(outFile, "%s ", name);

fprintf(outFile, "%d ", count);

fprintf(outFile, "%d ", credit);

fprintf(outFile, "\n");

}

fclose(outFile);

return 1;

}

int chkPoint(char card[][2], int maxDraw, int Pointt)

{

int c = maxDraw , APoint=0;

if (card[c][0] == 'A' )

{

if ((Pointt + 11 )> 21)

{

return 1;

}

else

{

return 11;

}

}

else if (card[c][0] == 'J' || card[c][0] == 'Q' || card[c][0] == 'K' || card[c][0] == '1')

{

return 10;

}

else

{

return atoi(&card[c][0]);

}

}

int drawCard(int i, char deck[CARDS][2], char deckSuit[CARDS][1])

{

int randN = 0;

int randS = 0;

int ckCard = 0;

char randNumber[13][2] = { "A","2","3","4","5","6","7","8","9","10","J","Q","K" };

char randSuit[4][1] = { "C","D","H","S" };

randN = (int)rand() % 12;

randS = (int)rand() % 3;

while (ckCard <= i)

{

if (randNumber[randN][0] == deck[ckCard][0] && randSuit[randS][1] == deckSuit[ckCard][1])

{

randN = (int)rand() % 12;

randS = (int)rand() % 3;

ckCard = 0;

}

else if (randNumber[randN][0] != deck[ckCard][0] && randSuit[randS][1] != deckSuit[ckCard][1] && ckCard == i)

{

deck[i][0] = randNumber[randN][0];

deckSuit[i][1] = randSuit[randS][1];

return randN;

}

else

{

ckCard++;

}

}

}

int betting(int money)//betting

{

char bet[1000];

int b = 0;

bool check;

do

{

check = true;

printf("Enter an amount you would like to bet : ");

gets(bet);

for (int i = 0; i < strlen(bet); i++)

{

if (!isdigit(bet[i])) {

check = false;

break;

}

}

b = atoi(bet);

if (b > money)

{

printf("your money is not enough.\n");

}

else if (b == 0 )

{

printf("you have to bet more.\n");

}

} while ((b > money || b == 0) || check == false) ;

return b;

}

int checkAce(struct Dealer dealer ,struct Player player , int money)

{

char select[5];

if (dealer.cardDealer[0][0] == 'A' && player.money > money)

{

do

{

printf("Do you want to insurance ? (Enter 'y' or 'n') : ");

gets(select);

if (strcmp(select, "y") == 0 || strcmp(select,"Y") == 0)

{

player.money += money/2;

printf("Amount of money left: %d credits. \n", player.money);

return player.money;

}

else if(strcmp(select, "n") == 0 || strcmp(select, "N")==0)

{

return player.money;

}

} while (strcmp(select, "y") != 0 && strcmp(select, "n") != 0 && strcmp(select,"Y") != 0 &&strcmp(select,"N")!= 0);

}

}

int ranCardDealer(char DealerCard[][2], int count, int PointD, char deck[CARDS][2], char deckSuit[CARDS][1])

{

int ten = 0;

while (PointD < 17)

{

ten = drawCard(count, deck, deckSuit);

DealerCard[count][0] = deck[count][0];

if (ten == 9)

{

printf("Dealer Card 10%c\n", deckSuit[count][1]);

}

else

{

printf("Dealer Card %c%c\n", DealerCard[count][0], deckSuit[count][1]);

}

PointD += chkPoint(DealerCard, count,PointD);

}

return PointD;

}

int play(struct Player player, struct Dealer dealer,int money)

{

int i = 0, draw = 0, pts = 0, ptsD = 0, plus = 0, m = 0, pts2 = 0, sp = 0, ten = 0, ckv = 0;

char deck[CARDS][2];

char deckSuit[CARDS][1];

char ckbuf, choice[15],c[15];

int ckSpilt, card3 = 0 , card4=0;

srand(time(NULL));

for (draw = 0; draw < 4; draw++) //draw player's cards

{

ten = drawCard(i, deck, deckSuit);

if (draw == 0 || draw == 1 )

{

dealer.cardDealer[draw][0] = deck[i][0]; //draw dealer's card

if (draw == 0)

{

if (ten == 9)

{

printf("Dealer Card 10%c\n", deckSuit[draw][1]);

}

else

{

printf("Dealer Card %c%c\n", dealer.cardDealer[draw][0], deckSuit[draw][1]);

}

printf("Dealer's points: %d\n\n", chkPoint(dealer.cardDealer, 0,dealer.point));

}

ptsD += chkPoint(dealer.cardDealer, draw,dealer.point);

}

else

{

player.cardPlayer[draw][0] = deck[i][0];

if (ten == 9)

{

printf("Player Card 10%c\n",deckSuit[draw][1]);

}

else

{

printf("Player Card %c%c\n", player.cardPlayer[draw][0], deckSuit[draw][1]);

}

pts += chkPoint(player.cardPlayer, draw, player.point); //check the value and add it to player's points

player.point = pts;

}

i++;

}

printf("Player's points %d \n", player.point);

m = checkAce (dealer,player,money);

if (m != 1)

{

player.money = m;

}

dealer.point = ptsD;

if (player.point == 21)

{

printf("Congrats! You got a blackjack !!\n");

win++;

player.money += money \* 2.5;

ckbuf = getchar();

return player.money;

}

else if (player.point < 21)

{

do

{

if (ckv == 0)

{

ckbuf = getchar();

}

printf("\nType 'hit' to be dealt another card. Type 'stand' to hold. Type 'double' to draw one card and bet more credit. : ");

gets(tolower(choice));

if (strcmp(choice, "hit") == 0 || strcmp(choice,"Hit") == 0 )

{

ten = drawCard(i, deck, deckSuit);

player.cardPlayer[i][0] = deck[i][0];

if (ten == 9)

{

printf("Player Card 10%c\n", deckSuit[i][1]);

}

else

{

printf("Player Card %c%c\n", player.cardPlayer[i][0], deckSuit[i][1]);

}

pts += chkPoint(player.cardPlayer, i, player.point);

player.point = pts;

printf("Player Points %d \n", player.point);

if (player.point == 21)

{

printf("Congrats! You got 21!\n");

win++;

return player.money += money \* 2;

}

else if (player.point > 21)

{

printf("You bust\n");

printf("You lose\n");

return player.money;

}

i++;

}

else if (strcmp(choice, "double") == 0 || strcmp(choice,"Double") == 0)

{

if (player.money > money)

{

ten = drawCard(i, deck, deckSuit);

player.cardPlayer[i][0] = deck[i][0];

if (ten == 9)

{

printf("Player Card 10%c\n", deckSuit[i][1]);

}

else

{

printf("Player Card %c%c\n", player.cardPlayer[i][0], deckSuit[i][1]);

}

pts += chkPoint(player.cardPlayer, i, player.point);

player.point = pts;

printf("Player Points %d \n", player.point);

if (player.point == 21)

{

printf("Congrats! You got 21!\n");

win++;

return player.money += money \* 2;

}

else if (player.point > 21)

{

printf("You bust\n");

printf("You lose\n");

return player.money;

}

player.money -= money;

break;

}

else

{

printf("You can't doubledown\n");

}

}

else if (strcmp(choice, "stand") == 0 || strcmp(choice,"Stand") == 0 )

{

break;

}

ckv++;

}

while (strcmp(choice, "stand") != 0 && (choice != '\n' && choice != EOF));

if (dealer.point < 17)

{

dealer.point = ranCardDealer(dealer.cardDealer, i, dealer.point,deck,deckSuit);

}

printf("Dealer Points %d \n", dealer.point );

if (player.point < dealer.point && dealer.point <= 21)

{

printf("You lose\n");

return player.money;

}

else if (player.point == dealer.point)

{

printf("You tie\n");

return player.money += money;

}

else if(player.point > dealer.point && player.point <=21 || dealer.point >21)

{

printf("You Win!\n");

win++;

return player.money += money\*2;

}

}

}

int main()

{

struct Player player;

struct Dealer dealer;

FILE \*fp,\*fp1;

char name[50],name1[50];

int money = 0 ,count =0, selectt = 0 , moneyF=0 , winF = 0 ;

char again[10], ckBuf, repeat[10] ;

int select, select2;

char draw, ckbuf;

char filename[1000] = "rule.txt";

printf(">>>> Welcome to BlackJack Let's get rich!.<<<<\n");

do

{

printf("Please Select the choice:\n");

printf("Press 1.New Game\n 2.Load Game\n 3.Quit\n");

scanf("%d", &select);

ckBuf = getchar();

if (select == 1)

{

printf("Enter your name: ");

gets(player.name);

fp = fopen("userData.txt", "r");

if (fp == NULL)

{

printf("Error opening file\n");

exit(1);

}

while (fscanf(fp, "%s %d %d", name, &winF, &moneyF) != EOF)

{

if (strcmp(player.name, name) == 0)

{

break;

}

else

{

continue;

}

}

if (strcmp(player.name, name) == 0)

{

do

{

printf("this name is Exist!\n");

printf("1. Load Game 2.Enter new name\n");

scanf("%d", &select2);

ckBuf = getchar();

if (select2 == 1)

{

player.money = moneyF;

player.Win = winF;

selectt = 1;

break;

}

else

{

printf("Enter your name: ");

gets(player.name);

displayFile(filename);

player.money = 500;

selectt = 0;

break;

}

} while (select2 != 1 || select2 !=2);

break;

}

else if (strcmp(player.name, name) != 0 )

{

displayFile(filename);

player.money = 500;

selectt = 0;

break;

}

}

else if (select == 2)

{

printf("Enter your name: ");

gets(player.name);

fp = fopen("userData.txt", "r");

if (fp == NULL)

{

printf("Error opening file\n");

exit(1);

}

while (fscanf(fp, "%s %d %d", name, &winF, &moneyF) != EOF)

{

if (strcmp(player.name, name) == 0)

{

break;

}

else

{

continue;

}

}

if (strcmp(player.name, name) == 0)

{

player.money = moneyF;

player.Win = winF;

selectt = 1;

}

else

{

player.money = 500;

selectt = 0;

}

break;

}

else if(select == 3)

{

exit(0);

}

else

{

printf("Invalid number\n");

}

} while (select != 1 || select != 2);

do

{

dealer.point = 0;

player.point = 0;

if (count != 0)

{

do

{

if (player.money > money)

{

printf("Press y/n to repeat bet: ");

gets(repeat);

if ((strcmp(repeat, "Y") == 0 || strcmp(repeat, "y") == 0) )

{

system("cls");

player.money -= money;//betting

break;

}

else if (strcmp(repeat, "N") == 0 || strcmp(repeat, "n") == 0)

{

money = betting(player.money);

player.money -= money;//betting

break;

}

}

else

{

money = betting(player.money);

player.money -= money;//betting

break;

}

} while (strcmp(repeat, "Y") != 0 && strcmp(repeat, "y") != 0 && strcmp(repeat, "N") != 0 && strcmp(repeat, "n") != 0);

}

else

{

money = betting(player.money);

player.money -= money;//betting

}

if (money < player.money)

{

printf("Amount of money left: %d credits. \nGood luck!\n", player.money);

}

printf("\n");

player.money = play(player, dealer, money);

count++;

do

{

printf("Press Y/N to play again : ");

gets(again);

} while (strcmp(again, "Y") != 0 && strcmp(again, "N") != 0 && strcmp(again, "y") != 0 && strcmp(again, "n") != 0);

} while (strcmp(again,"Y")==0 || strcmp(again, "y") == 0 || player.money <= 0 );

player.Win = win;

if (selectt == 0)

{

writeFile(player.name, player.Win, player.money);

}

else if (selectt == 1)

{

updateFile(name1,winF,moneyF,player);

}

system("pause");

}

**13016235: C Programming**

**First Semester, 2018**

# Project Proposal

1. Project developer

|  |  |
| --- | --- |
| Student ID | Name |
| 61090004 | Chanyanuch Likitpanjamanon |

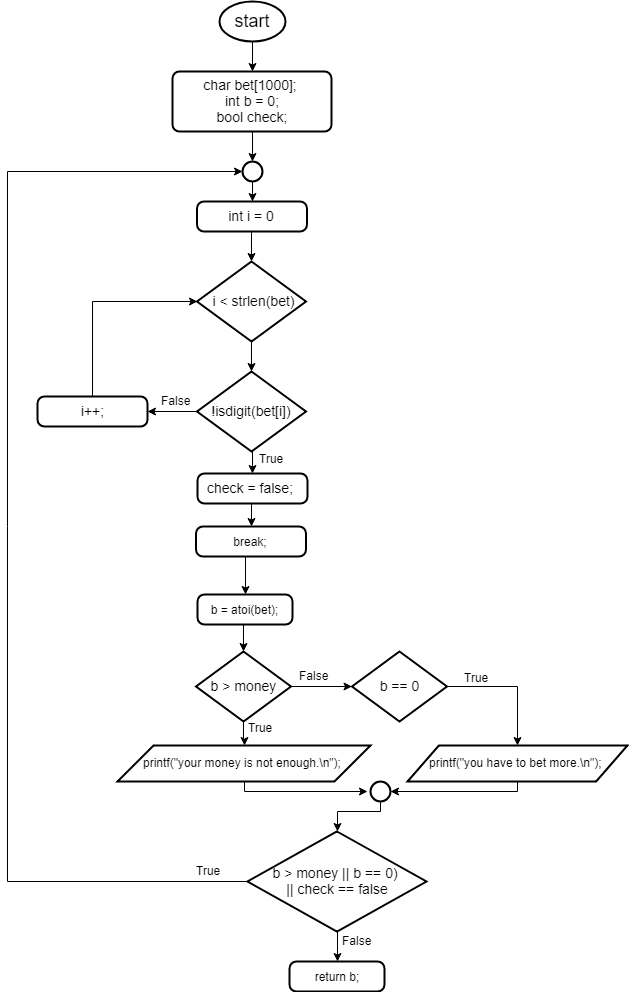
1. Project title

|  |
| --- |
| Blackjack : Let’s get rich |

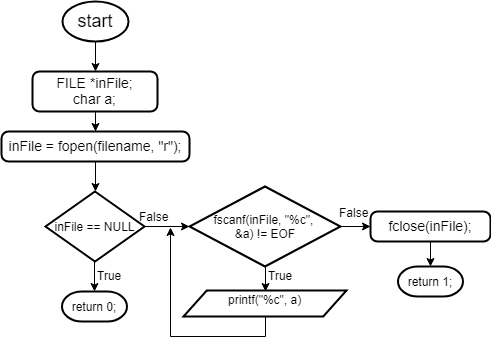
1. Project description and requirements

|  |
| --- |
| **Project description:**  The concept of this project is like the Blackjack or twenty-one card games. The objective of this game is that the player has to beat the dealer in one of the following ways:   * Get 21 points on the player's first two cards without a dealer get 21 points * Reach a final score higher than the dealer without exceeding 21 * Let the dealer draw additional cards until their hand exceeds 21   The value of cards two through ten is their value (2 through 10). Jack, Queen, and King are all worth ten. Aces are worth one or eleven.  At the beginning of the game, the player and the dealer will get two cards randomly and the player will receive 1000 credits. After receiving an initial two cards, the player has seven standard options: "hit", "stand", "double down", "split", “surrender”, ”repeat” and “insurance”.   * **Hit:** Draw one card from the deck * **Repeat:** Repeat the bet * **Stand:** Take no more cards * **Double down:** The player is allowed to increase the initial bet by up to 100% and the player can draw only one card * **Split**: If the first two cards of a hand have the same value, the player can split them into two hands * **Surrender**: (only available as first decision of a hand): When the player surrenders, the house takes half the player's bet and returns the other half to the player. * **Insurance**: Insurance: If the dealer's up card is an ace, the player is offered the option of taking "insurance" before the dealer checks the hole card. If the player wants to use insurance, the player has to pay half of the credits that the player bets. If the dealer gets Blackjack, the player will gain the insurance money. However, if the dealer loses the game, player will get all insurance money and the dealer’s money.   If the player runs out of credits, the game will be over. The player can then restart and play again. When the game is over the console will show the biggest win that the player gains from the dealer and the total amount of wins.  **Project requirements:**  This game will use the standard library such as <stdio.h>, <stdlib.h>, <string.h> , <time.h>, etc. Moreover, it will use the file to store the data of the player. When the game is over, it will show the biggest win and total credits that the player received from the dealer. The game uses array to store the deck of cards. At the beginning of the game, the random method will be used to random four cards from the deck. |

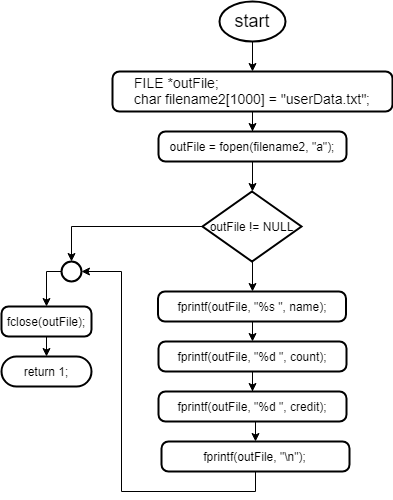
**int betting(int money)**

****

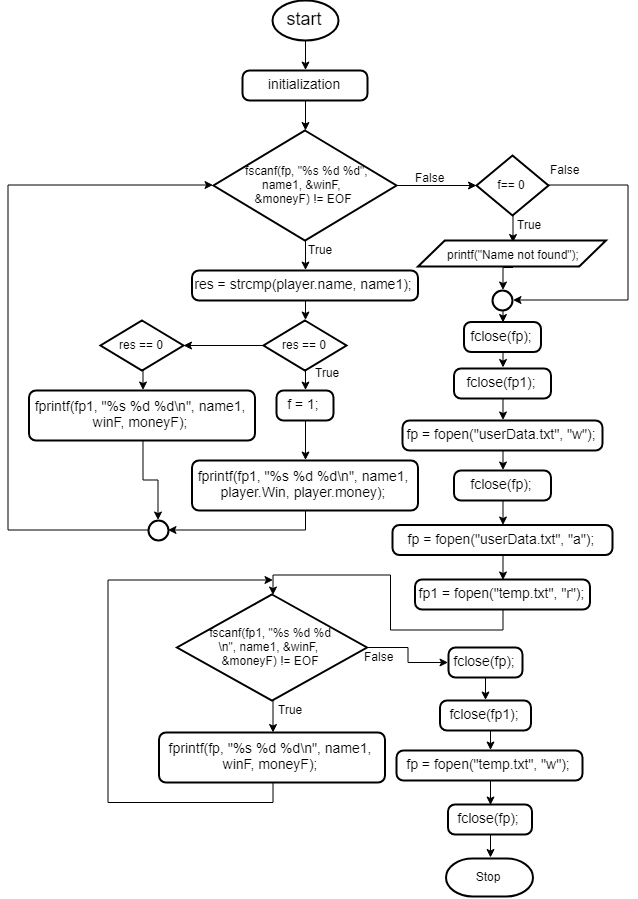
**int displayFile(char \*filename)**

****

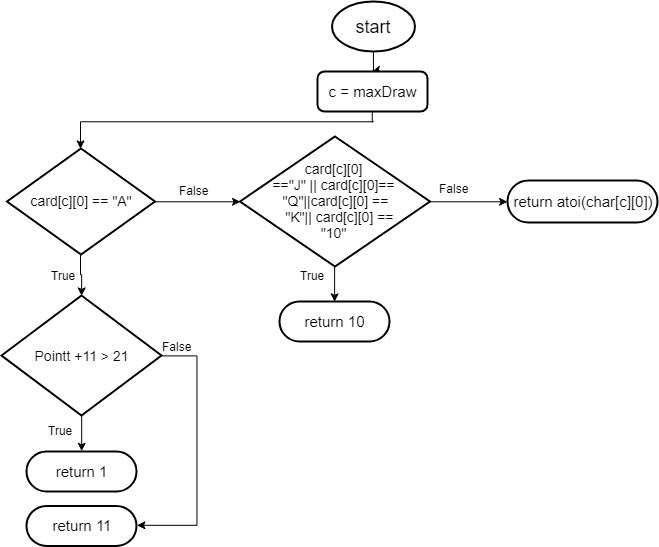
**int writeFile(char \*name,int count,int credit)**

****

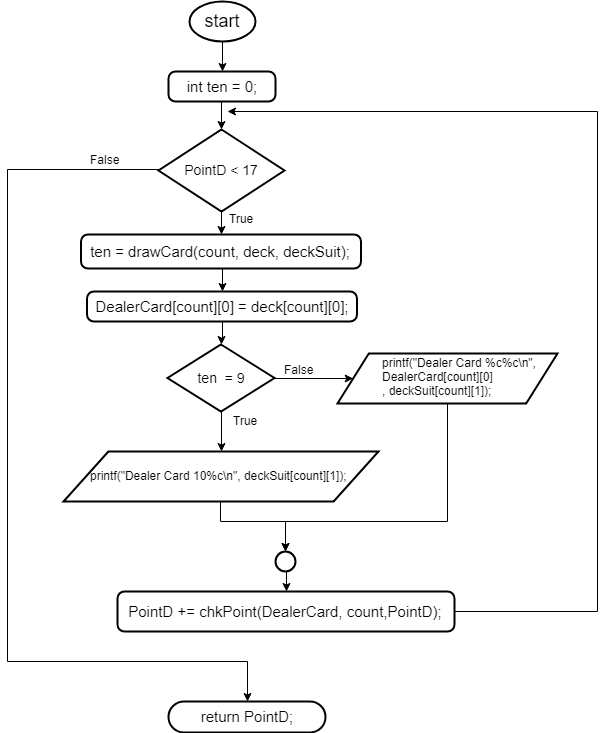
**int updateFile(char name1[50] , int winF , int moneyF , struct Player player)**



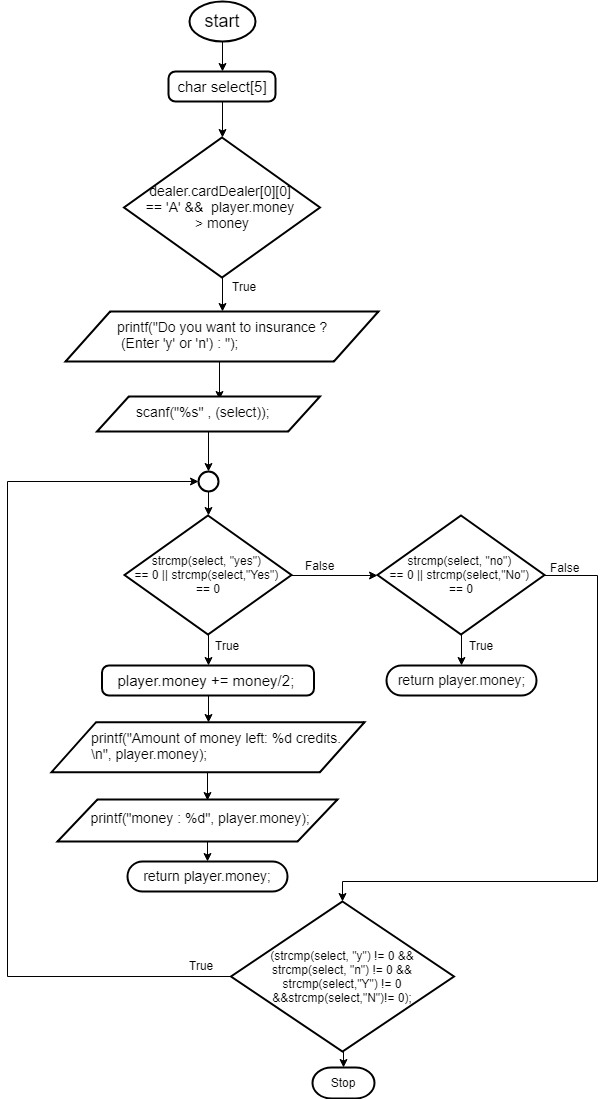
int chkPoint(char card[][2], int maxDraw, int Pointt)



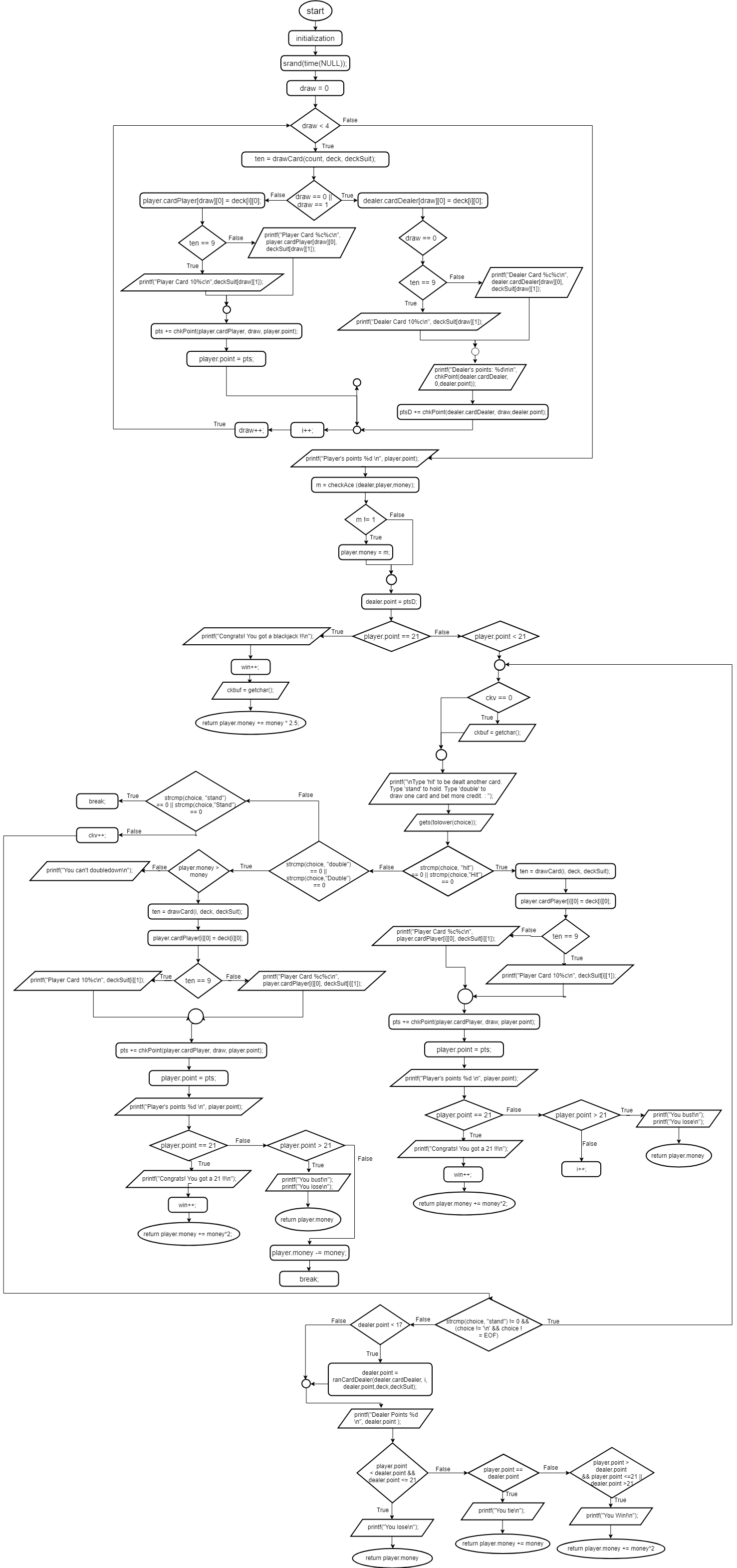
int ranCardDealer(char DealerCard[][2], int count, int PointD, char deck[CARDS][2], char deckSuit[CARDS][1])

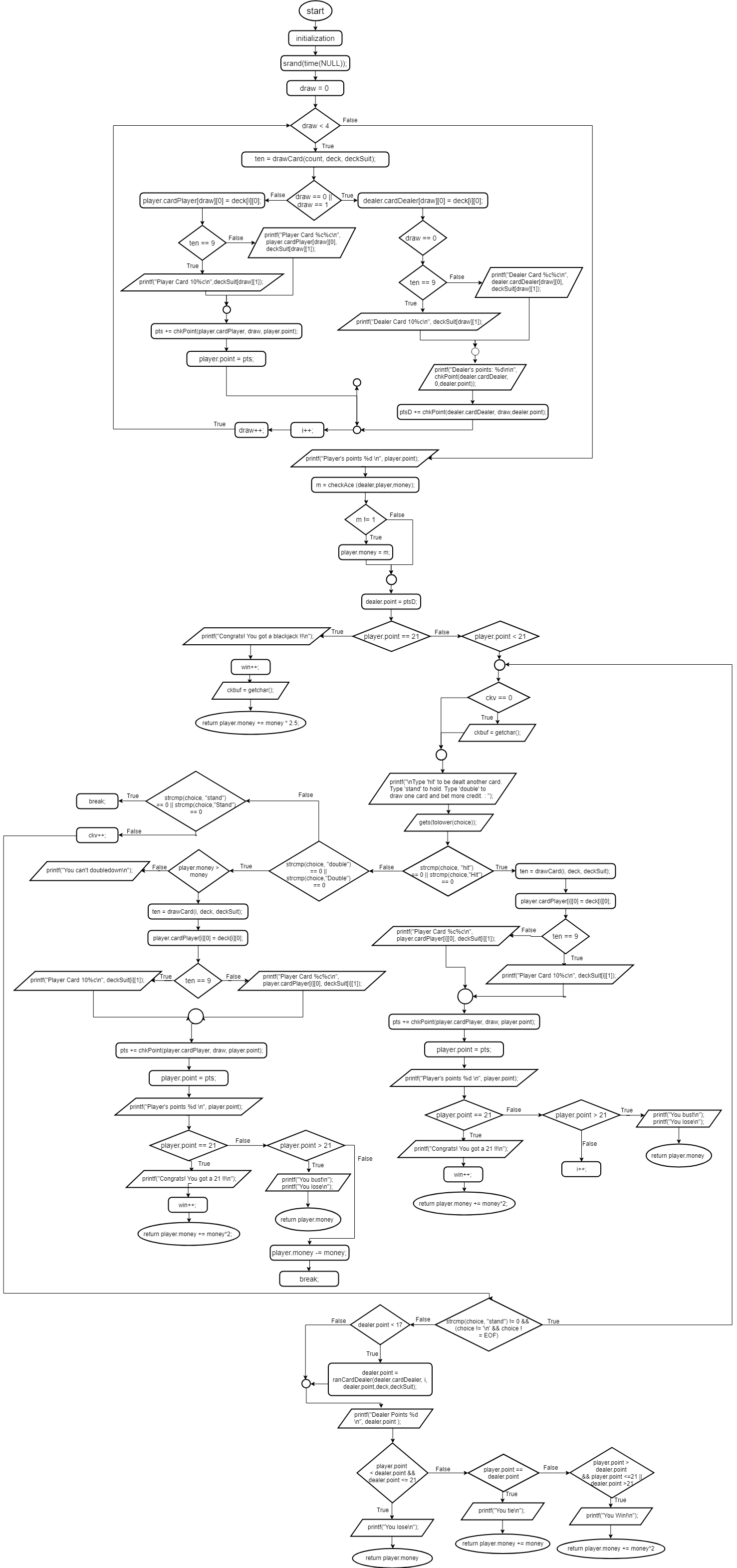


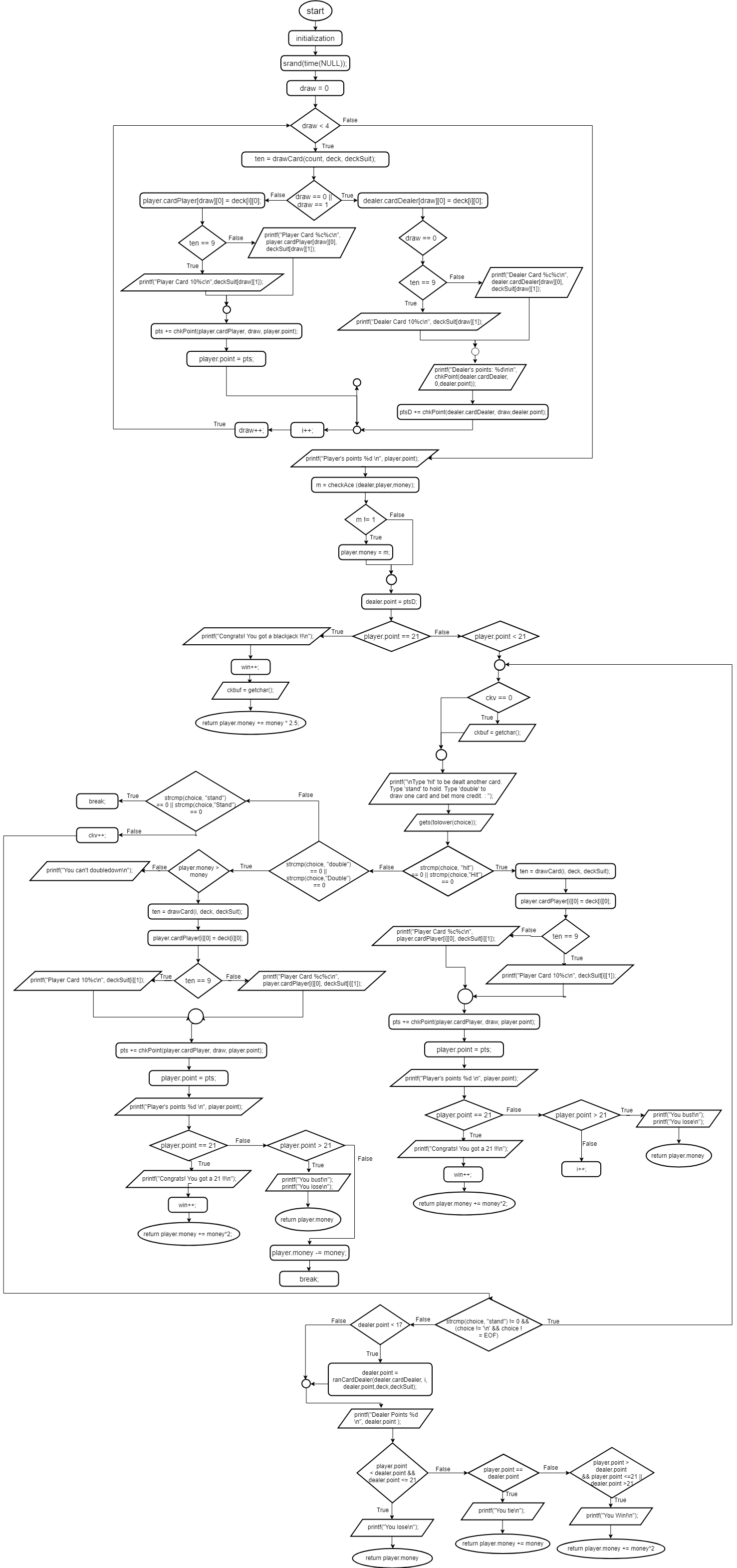
int checkAce(struct Dealer dealer ,struct Player player , int money)

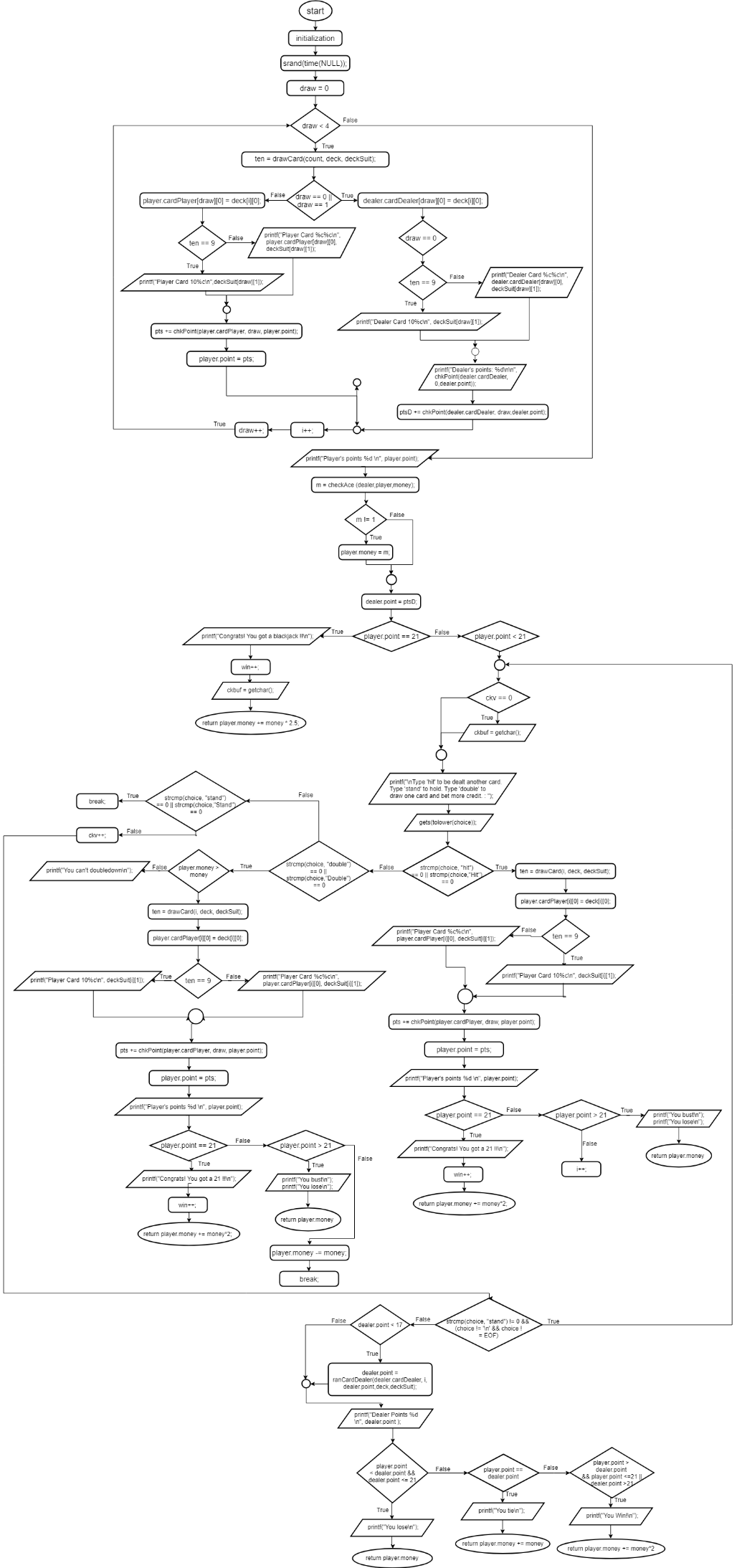


int play(struct Player player, struct Dealer dealer,int money)

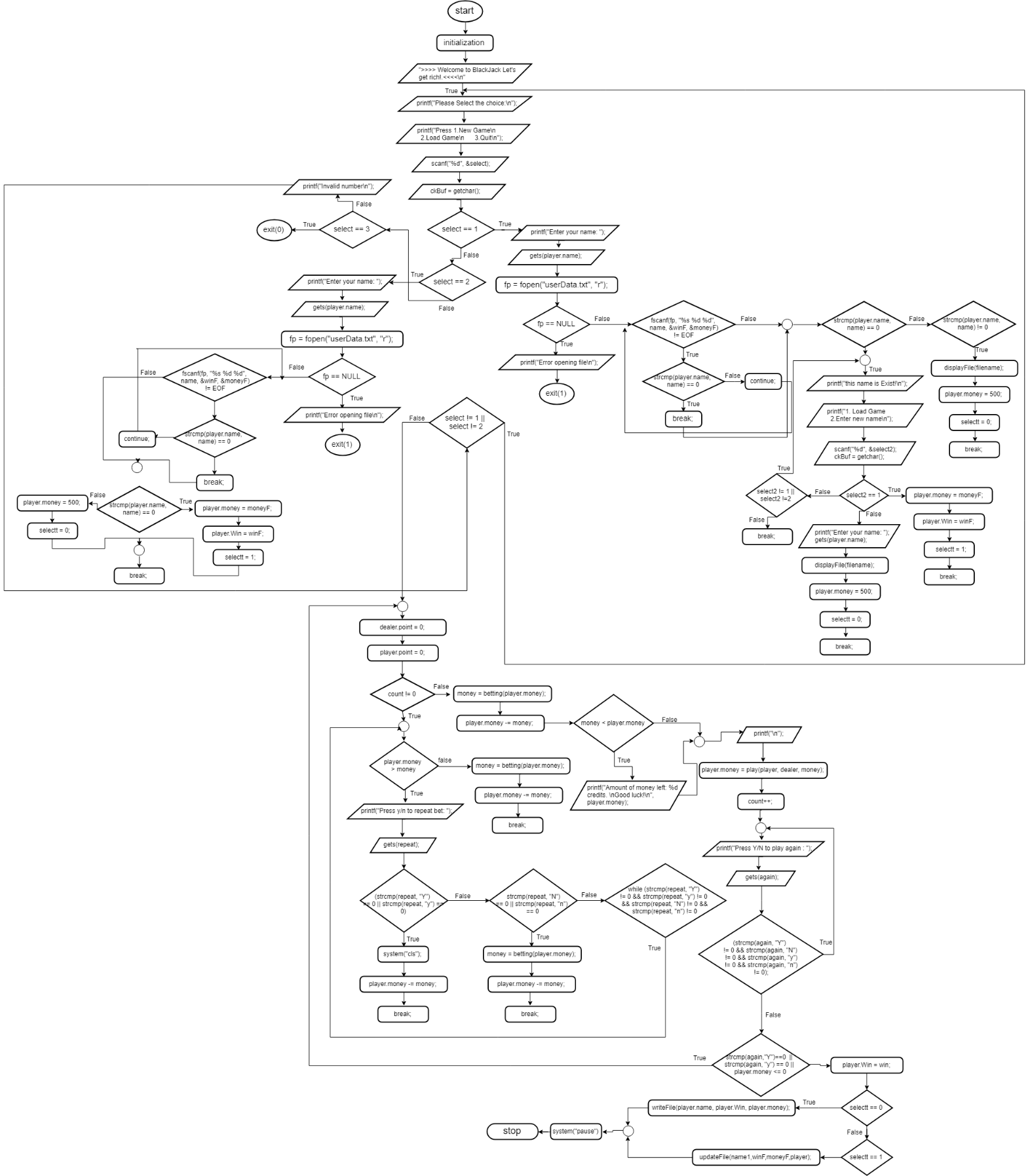


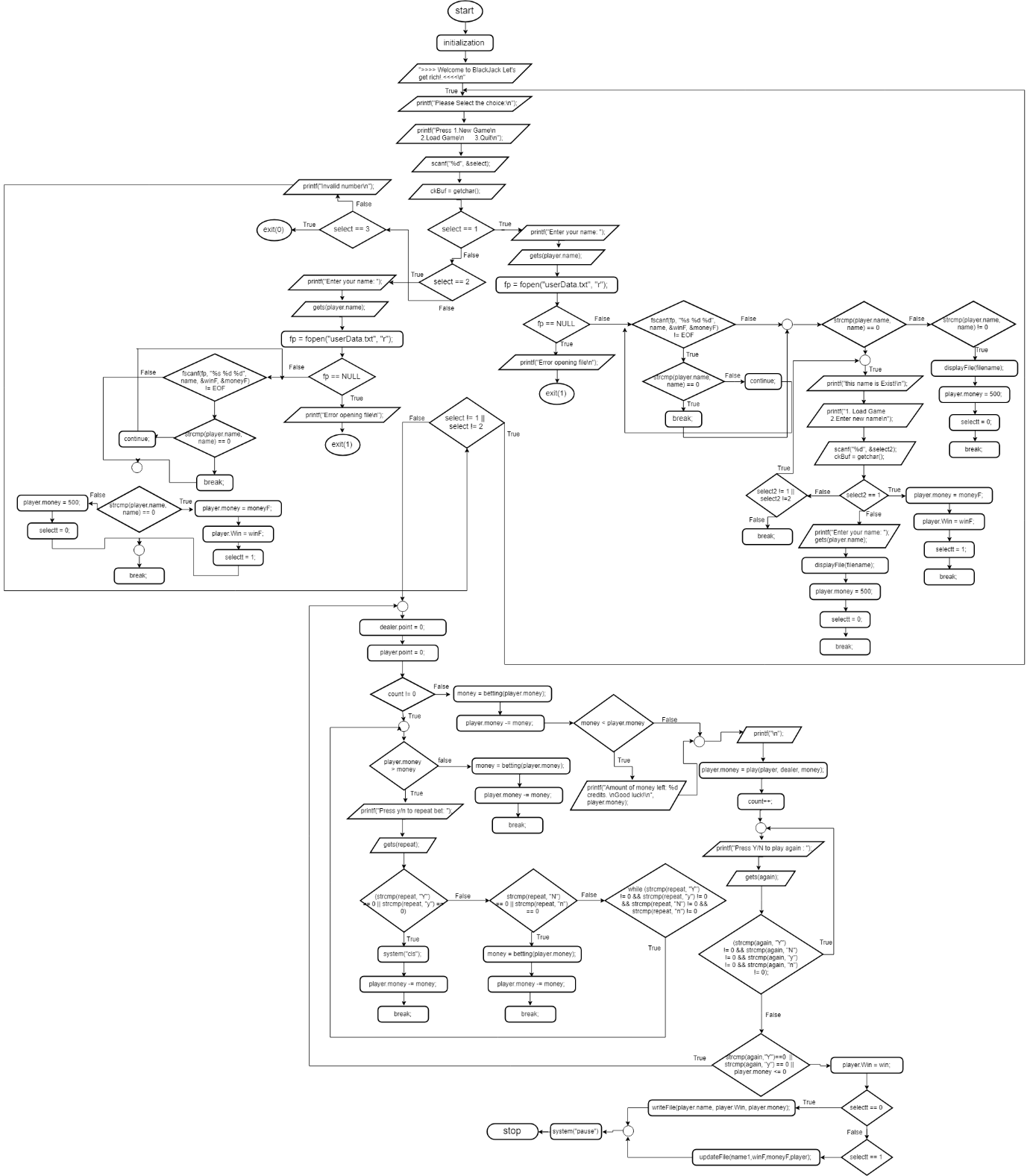


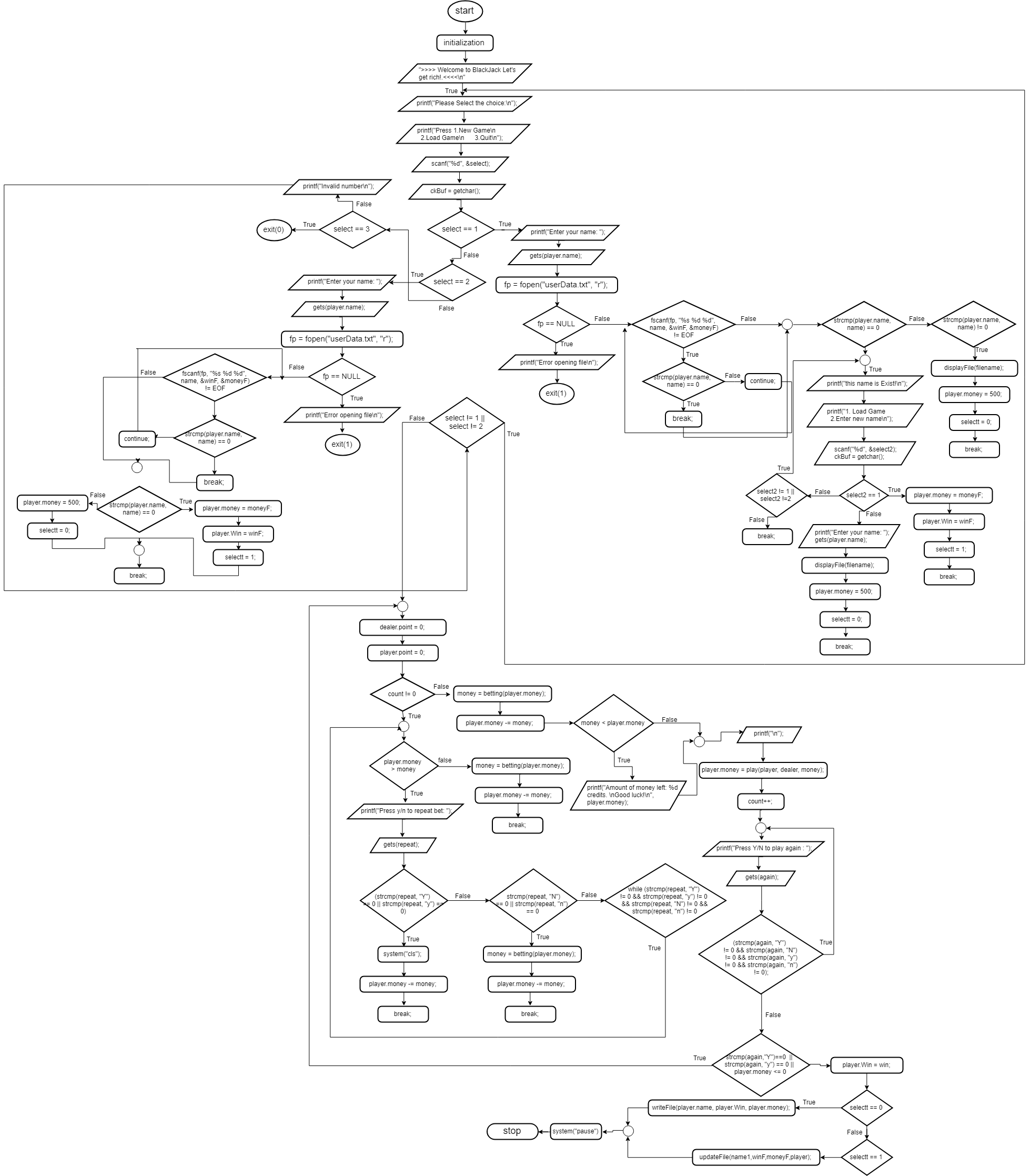




int main()







int drawCard(int i, char deck[CARDS][2], char deckSuit[CARDS][1])

