```
                          ┌─────────────┐
                          │    Start    │
                          └──────┬──────┘
                                 │
                          ┌──────▼──────┐
                          │ srand(time(0)) │
                          └──────┬──────┘
                                 │
              ┌──────────────────▼──────────────────┐
              │         Declared Variables:         │
              │  Card cards, Deck deck, Player *p,   │
              │ fstream rules, char *filename, string readLine, │
              │ int numOfP, int i, int j, int a=b=c=d, int index │
              └──────────────────┬──────────────────┘
                                 │
              ┌──────────────────▼──────────────────┐
              │        Initialized Variables:       │
              │       *fileName="Rules.txt",        │
              │          a=b=c=d=2, index=51        │
              └──────────────────┬──────────────────┘
                                 │
                    ┌────────────▼────────────┐
                    │ rules.open(fileName, ios::in) │
                    └────────────┬────────────┘
```

Flowchart for a card game program.

Start → srand(time(0))

Declared Variables:
Card cards, Deck deck, Player *p, fstream rules, char *filename, string readLine, int numOfP, int i, int j, int a=b=c=d, int index

Initialized Variables:
*fileName="Rules.txt", a=b=c=d=2, index=51

rules.open(fileName, ios::in)

Decision: `getline(rules, readLine` — **True** → `cout << readLine <<endl;` (loops back), **False** →

`cout << "How many players are playing? Only 3 can play at the " << "same time plus the dealer." << endl;`

Decision: `numOfP<1 || numOfP>3` — **True** → Decision: `numOfP==0`
- `numOfP==0` **True** → `cout <<"There's no one to play with! Try again." << endl; cin>>numOfP;`
- `numOfP<0` **True** → `cout<<"Cannot enter negative numbers, try again."<<endl; cin>>numOfP;`
- `numOfP>3` **True** → `cout <<"That's too many people! Try again." << endl; cin>>numOfP;`

Decision: `numOfP<1 || numOfP>3` — **False** →

`p = new Player[4];`

`cout << "Enter name for the players." << endl;`

Decision: `i=1;i<=numOfP;i++` — **True** → `cin >> p[i].name` (loops back), **False** →

(1)

```
                                    ( 1 )
                                      │
                                      ▼
                        ┌──────────────────────────┐
                        │    p[0].name="Dealer";    │
                        └──────────────────────────┘
                                      │
                                      ▼
False               ┌──┬──────────────────────┬──┐
                    │  │   SetupCards(deck)    │  │
                    └──┴──────────────────────┴──┘
                                      │
                                      ▼●
True         ┌──────────────┐     True
┌──────────────┐◄── ╱ j=0;j<2;j++ ╲ ◄── ╱ i=0;i<=numOfP;i++ ╲
│ p[i].hand[j] =│    ╲            ╱      ╲                  ╱
│ draw(deck,index)│   ╲          ╱        ╲                ╱
└──────────────┘     └──────────┘           │
       │                                     ▼
       └─────────────────►                 True
                                    ╱ numOfP>=1 ╲ ──► ╱ j=0;j<b;j++ ╲ true ┌──────────────────────┐
                                    ╲           ╱     ╲             ╱ ───► │ PrintCard(p[1].hand[j]);│
                                      │              False └──────────────┘
                                      ▼                                    
                                    True                                   
                                    ╱ numOfP>=2 ╲ ──► ╱ j=0;j<b;j++ ╲ true ┌──────────────────────┐
                                    ╲           ╱     ╲             ╱ ───► │ PrintCard(p[1].hand[j]);│
                                      │              False └──────────────┘
                                      ▼                                    
                                    True                                   
                                    ╱ numOfP>=3 ╲ ──► ╱ j=0;j<b;j++ ╲ true ┌──────────────────────┐
                                    ╲           ╱     ╲             ╱ ───► │ PrintCard(p[1].hand[j]);│
                                      │  False          False └──────────────┘
                                      ▼
                        ┌──┬──────────────────────────────┬──┐
                        │  │ Dealer(deck,p,numOfP,index,a,b,c,d) │  │
                        └──┴──────────────────────────────┴──┘
                                      │
                                      ▼
                        ┌──┬──────────────────────────┬──┐
                        │  │  CheckSum(p,numOfP,b,c,d); │  │
                        └──┴──────────────────────────┴──┘
                                      │
                                      ▼
                        ┌──┬──────────────────────┬──┐
                        │  │  CheckWin(p,numOfP)   │  │
                        └──┴──────────────────────┴──┘
                                      │
                                      ▼
                        ┌──────────────────────────┐
                        │       delete []p;         │
                        │       return 0;           │
                        └──────────────────────────┘
                                      │
                                      ▼
                                  (  End  )
```

void PrintCard(Card c)

```
void PrintCard(Card c)
    ↓
│ PrintValue(c); │
    ↓
/ cout << " of "; /
    ↓
│ PrintSuit(c); │
    ↓
/ cout << endl; /
    ↓
  end
```

void PrintValue(Card c)

```
void PrintValue(Card c)
    ↓
c.name==CardName::Ace ──True──> / cout<<"Ace"; / ──> c.value=11 ──┐
    │ False                                                        │
    ↓                                                              │
c.name==CardName::Jack ──True──> / cout<<"Jack"; / ──> c.value=10 ─┤
    │ False                                                        │
    ↓                                                              │
c.name==CardName::Queen ──True──> / cout<<"queen"; / ──> c.value=10 ┤
    │ False                                                        │
    ↓<─────────────────────────────────────────────────────────────┘
│ cout << c.value; │
    ↓
  End
```

void PrintSuit(Card c)

```
void PrintSuit(Card c)
    ↓
c.suit==Suits::Clubs ──True──> / cout<<"Ace"; / ──┐
    │ False                                        │
    ↓                                              │
c.suit==Suits::Diamonds ──True──> / cout<<"Ace"; / ┤
    │ False                                        │
    ↓                                              │
c.suit==Suits::Hearts ──True──> / cout<<"Ace"; / ──┤
    │ False                                        │
    ↓                                              │
c.suit==Suits::Spades ──True──> / cout<<"Ace"; / ──┤
    │ False                                        │
    ↓<─────────────────────────────────────────────┘
  End
```

void shuffle(Deck &d)

```
int n,m;
Card temp=d.arrCards[0];
```

n=0;n<52;n++ — True →

```
m=rand()%52;
temp=d.arrCards[n];
d.arrCards[n]
d.arrCards[m]
d.arrCards[n]=d.arrCards[m];
d.arrCards[m]=temp;
```

False → **End**

void PrintAll(Deck d)

int col=(int)Suits::Clubs; col<=(int)Suits::Spades; col++ — True →

False → **End**

int row=(int)CardName::Ace; row <= (int)CardName::King; row++ — False

True →

```
int index=
(13*col)+row-1;
```

```
PrintCard(d.arrCards[index]);
```

```
void SetupCards(Deck &d)
```

int col=(int)Suits::Clubs; col<= (int)Suits::Spades; col++

True → int row=(int)CardName::Ace; row <= (int)CardName::King; row++

False (loop back)

False → shuffle(d);

End

True →
```
Card c;
c.suit = (Suits)col;
c.name = (CardName)row
```

c.name==CardName::Jack

True → c.value=10;

flase ↓ True

c.name==CardName::Queen

True → c.value=10;

false ↓ True

c.name==CardName::Jack

True → c.value=10;

false ↓

```
c.value = (int)c.name;
```

```
int index = (13*col)+row-1
d.arrCards[index]=c
```

```
Card draw(Deck &d, int &index)
```

```
return d.arrCards[index--];
```

End

```
void Dealer(Deck &deck, Player *p,int numOfP, int &index,int &a, int &b, int &c, int &d)
```

```
int choice,i,j;
```

True → cout << "Press 1 to let player one draw."<

numOfP>=1

False

True → cout << "Press 2 to let player two draw."<

numOfP>=2

False

True → cout << "Press 3 to let player three draw."<

numOfP>=3

False

```
cout <<"Press 4 to
continue."<> choice;
```

choice!=4 ← (2)

False → (3)

choice==1 —**False**→ choice==2 —**False**→ choice==3

True | **True** | **True**

```
p[1].hand[b] =
draw(deck, index);
        b++;
```

```
p[2].hand[c] =
draw(deck, index);
        c++;
```

```
p[3].hand[d] =
draw(deck, index);
        d++;
```

(1)

```
(1)

numOfP>=1  --True-->  j=0;j<b;j++  --true-->  PrintCard(p[1].hand[j]);
   |                       |
   |                     False
   v
numOfP>=2  --True-->  j=0;j<b;j++  --true-->  PrintCard(p[1].hand[j]);
   |                       |
   |                     False
   v
numOfP>=3  --True-->  j=0;j<b;j++  --true-->  PrintCard(p[1].hand[j]);
   |                       |
 False                   False
   v
cout << "Do you want to draw more cards?"<

   cout << "Press 1 to let    <--True--  numOfP>=1
   player one draw."<                        |
                                           False
                                             v
   cout << "Press 2 to let    <--True--  numOfP>=2
   player two draw."<                        |
                                      True   False
                                             v
   cout << "Press 3 to let    <--True--  numOfP>=3
   player three draw."<                      |
                                           False
                                             v
                              cout <<"Press 4 to continue."<> choice;

(2)

(3) --> int total=0; --> cout<<endl<<endl<<p[0].name<<endl; --> i=0;i<a;i++  --False-->  p[0].total=total;
                                                                   |                           |
                                                                 True                          v
                                                                   v                          (4)
                                              PrintCard(p[0].hand[i]);
                                              total+=p[0].hand[i].value;
```

Flowchart

(4) → Decision: `p[0].total<17`

- **False** → `cout << p[0].name <<" has a value of " << total << endl;` → **End**
- **True** →
  ```
  p[0].hand[a] = draw(deck, index);
  PrintCard(p[0].hand[i])
  total+=p[0].hand[a].value;
  p[0].total=total;
  a++;
  ```
  (loops back to decision `p[0].total<17`)

---

void CheckSum(Player *p, int numOfP, int b, int c, int d)

```
int i;
int total=0;
```

Decision: `numOfP>=1`

- **True** → `p[1].total=0;` → Decision: `i=0;i<b;i++`
  - **True** → `total+=p[1].hand[i].value;` (loops back)
  - **False** → `cout <<endl<< p[1].name <<" has a value of " << total <<endl<<endl;` →
    ```
    p[1].total=total;
    total=0;
    ```
- **False** → Decision: `numOfP>=2`
  - **True** → `p[2].total=0;` → Decision: `i=0;i<c;i++`
    - **True** → `total+=p[2].hand[i].value;` (loops back)
    - **False** → `cout <<endl<< p[2].name <<" has a value of " << total <<endl<<endl;` →
      ```
      p[2].total=total;
      total=0;
      ```
  - **False** → Decision: `numOfP>=3`
    - **True** → `p[3].total=0;` → Decision: `i=0;i<di++`
      - **True** → `total+=p[3].hand[i].value;` (loops back)
      - **False** → `cout <<endl<< p[3].name <<" has a value of " << total <<endl<<endl;` →
        ```
        p[3].total=total;
        total=0;
        ```
    - **False** → **End**

```
void CheckWin(Player *p,int numOfP)
```

False ← `int i=1;i<=numOfP;i++` → End

True

`p[i].total<=21` — False → `cout<< p[i].name <<"'s hand busted! "<< " is out of the game."<<endl;`

True

`p[0].total<p[i].total || p[0].total>21` — True → `cout << p[i].name <<" wins!" << endl;`

False

`p[0].total==p[i].total` — True → `cout << "Its a tie with the dealer with " << p[i].name <<"."<< " Play Again!"<<endl;`

False

`cout <<"The dealer wins against " <<p[i].name << "."<< endl;`