

```
//
// MN21.h
// MNCards
//
// Created by Moo on 6/12/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import <UIKit/UIKit.h>
@class MN21Lane;
@class MNCard;
@class MNDeck;

@interface MN21 : NSObject {
    NSArray *lanes;
    MNDeck *deck;
    long int score;
    int decks;
}

@property (readonly) long int score;

- (id)init;
- (id)initWithNumberOfDecks:(int)numberOfDecks;
- (NSError*)addCard: (MNCard *)card toLane:(int)laneNumber;
- (void)gameOver;
- (void)resetLanes;
- (int)valueOfLane:(int)lane;
- (MNCard *)nextCard;
- (int)cardsInDeck;
- (void)newGame;
- (NSString *)historyOfLane:(int)lane;

@end
//
// MN21Lane.h
// MNCards
//
// Created by Moo on 6/12/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import <UIKit/UIKit.h>

@class MNCard;
@interface MN21Lane : NSObject {
    int value;
    int laneNumber;
    int numberOfCards;
}
```

```

    BOOL containsAce;
    NSMutableArray *history;
}

- (id)init;

- (NSError*)addValue:(MNCard *)card;
- (void)resetLane;
- (NSString *)historyString;

@property (readonly) int value, laneNumber;
@end
//
// MNCard.h
// MNCards
//
// Created by Moo on 6/11/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import <UIKit/UIKit.h>

extern NSString * const SPADES;
extern NSString * const CLUBS;
extern NSString * const DIAMONDS;
extern NSString * const HEARTS;

enum {
    ACE = 1, JACK = 11, QUEEN, KING
} fValues;

@interface MNCard : NSObject {
    int serialNumber;
    NSString* suit;
    int numberValue;
    int faceValue;
    UIImage *frontImage;
}

+ (NSString *)faceValueToCardName:(int)value; //can be enum'd value

- (id)initWithId:(int)uid suit:(NSString *)theSuit faceValue:(int)fValue
    frontImage:(UIImage *)image andNumberValue:(int)numValue;

- (NSString *)description;
- (void)dealloc;

@property (readonly) int serialNumber, numberValue, faceValue;
@property (readonly, copy) NSString *suit;

```

```

@property (readonly) UIImage *frontImage;
@end
//
// MNDeck.h
// MNCards
//
// Created by Moo on 6/11/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import <UIKit/UIKit.h>
@class MNCard;

typedef enum{ //This isn't Implemented Yet!
    RummyMode, PyramidMode
} MNCardValueMode;

@interface MNDeck : NSObject {
    int size;
    NSMutableArray *deck;
    int numberOfDecks;
    int mode;
}

- (id)initWithNumberOfDecks:(int)numDecks andMode:(MNCardValueMode)modez;
+ (NSMutableArray *)createDeck:(int)numDecks andMode:(MNCardValueMode)modez;
- (void)shuffle;
- (MNCard *)draw;
- (NSMutableArray *)dealCards:(int)numCards;
- (int)count;

@property (readonly) NSMutableArray *deck;
@property (readonly) int size;
@end
//
// NSMutableArray+Shuffle.h
// MNCards
//
// Created by Moo on 6/12/09.
// Copyright 2009 Micmoo. All rights reserved.
//
// This is an Obj-c Catagory, it extends functionality of existing class
#import <UIKit/UIKit.h>
@interface NSMutableArray (Shuffling)
- (void)shuffle;
@end

//

```

```

// Prefix header for all source files of the 'Run 21' target in the 'Run 21' project
//
#import <Availability.h>

typedef enum { //These are my different possible Errors / return values.
    EverythingIsOK = 0, LaneIsOver21, BadLaneNumber, NotEnoughCardsInDiscardPile,
    NotEnoughCardsInDeck, NoCardInHand,
    FiveCardScore, TwentyOne = 21
} MNErrors;

#ifdef __OBJC__
#import <Foundation/Foundation.h>
#import <UIKit/UIKit.h>
#endif

//
// MN21.m
// MNCards
//
// Created by Moo on 6/12/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import "MN21.h"
#import "MN21Lane.h"
#import "MNCard.h"
#import "MNDeck.h"

@implementation MN21
@synthesize score;

//Meat of the Run21 Game.

- (id)init { //shortcut
    return [self initWithNumberOfDecks:1];
}

- (id)initWithNumberOfDecks:(int)numberOfDecks {
    self = [super init];
    //Fill lane array with lane Objects
    lanes = [[NSArray alloc] initWithObjects:@"Lane Array",
        [[MN21Lane alloc] init],[[MN21Lane alloc] init],
        [[MN21Lane alloc] init],[[MN21Lane alloc] init],[[MN21Lane alloc] init],nil
    ];
    deck = [[MNDeck alloc] initWithNumberOfDecks:numberOfDecks andMode:0];
}

```

```

    decks = numberOfDecks;
    [deck shuffle];
    return self;
}

- (MNErrror)addCard: (MNCARD *)card toLane:(int)laneNumber {
    if (laneNumber > 5 || laneNumber < 1) { //We only have 5 lanes
        return BadLaneNumber;
    }

    MNErrror e = [[lanes objectAtIndex:laneNumber] addValue:card]; // Adds card, returns
status
    [card release];
    if (e == LaneIsOver21) { //if were over 21, dec. score
        [[lanes objectAtIndex:laneNumber] resetLane];
        score--;
        return e;
    }
    if (e == TwentyOne){ //If we got 21 we ++ score!!!
        score++;
        return EverythingSOK;
    }

    if (e == FiveCardScore) { //If they have 5 cards in a lane we also ++ score
        score++;
        [[lanes objectAtIndex:laneNumber] resetLane];
    }
    return EverythingSOK;
}

- (void)resetLanes {
    for (int i = 1; i < 6; i++){
        [[lanes objectAtIndex:i] resetLane];
    }
}

- (int)cardsInDeck {
    return [deck count];
}

- (int)valueOfLane:(int)lane {
    return (int)[[lanes objectAtIndex:lane] value];
}

- (MNCARD *)nextCard {
    return [deck draw];
}

- (void)gameOver {

```

```

    [self resetLanes];
}

- (NSString *)historyOfLane:(int)lane{
    return [[lanes objectAtIndex:lane] historyString];
}

- (void)newGame {
    if(deck){
        [deck release];
        deck = nil;
    }
    deck = [[MNDeck alloc] initWithNumberOfDecks:decks andMode:0];
    [deck shuffle];
    score = 0;
    [self resetLanes];
}

@end
//
// MN21Lane.m
// MNCards
//
// Created by Moo on 6/12/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import "MN21Lane.h"
#import "MNCard.h"

@implementation MN21Lane
@synthesize value, laneNumber;

- (id)init{
    self = [super init];
    value = 0;
    numberOfCards = 0;
    history = [[NSMutableArray alloc] init];
    return self;
}

- (MNErrors)addValue:(MNCard *)card{
    // Takes a MNCard and takes the number value. Computes how to add the
    // value to the current lane.

    int val = [card numberValue];
    int faceValue = [card faceValue];
    numberOfCards++;
}

```

```

if (val == ACE){
    if (value + 11 == 21){
        [self resetLane];
        return TwentyOne;
    }
    if (11 + value < 22){ //if the value with 11 added is less than 22
        value += 11; //We can add 11 for the time being, but if the users
        containsAce = YES; // Total ends up being to high next turn,
        val = 11;
        //We set this value to let us evalute the ace as 1 instead if we need to.
    }

    else value += 1; //If 11 doesn't work, add 1
}

else {
    if ((val + value) > 21){ // If our cards end up being greater then 21...
        if(containsAce){ // If we have an ace, see if we can convert the value from
11 -> 1
            if ((val + value - 10) < 22){// if werestill under 22 when we change
from 11 -> 1 (a diff of 10)
                value -= 10; // subtract 10 from the value
                containsAce = NO; //we no longer can use this ace as a lower number.
            }

            else { //if its to high even with the adjusted ace, you lose.
                return LaneIsOver21;
            }
        }

        else { //if we don't have an ace, were hopeless, and die.
            return LaneIsOver21;
        }
    }

    value += val; // if we are less then 21 (we passed the above test)
}
if (numberOfCards == 5) return FiveCardScore; // if we have 5 cards in a lane, we
win
if (value == 21) { // or if we are == to 21, we win.
    [self resetLane];
    return TwentyOne;
}

[history addObject:[MNCard faceValueToCardName:faceValue]];
return EverythingsoK; // Else keep going.
}

- (NSString *)historyString{

```

```

    return [history componentsJoinedByString:@"\n"];
}

- (void)resetLane{
    value = 0;
    numberOfCards = 0;
    [history removeAllObjects];
}

@end
//
// MNCard.m
// MNCards
//
// Created by Moo on 6/11/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import "MNCard.h"

NSString * const SPADES = @"Spades";
NSString * const CLUBS = @"Clubs";
NSString * const DIAMONDS = @"Diamonds";
NSString * const HEARTS = @"Hearts";

@implementation MNCard
@synthesize serialNumber,suit,faceValue,numberValue,frontImage;

- (id)initWithId:(int)uid suit:(NSString *)theSuit faceValue:(int)fValue
    frontImage:(UIImage *)image andNumberValue:(int)numValue
{
    self = [super init];
    suit = [theSuit copy];
    faceValue = fValue; //ex 13 = King
    numberValue = numValue; // ex King = 10
    serialNumber = uid;
    frontImage = [image retain];
    return self;
}

+ (NSString *)faceValueToCardName:(int)value{ //Takes Facevalue (enum), returns String
    switch (value) {
        case KING:
            return @"King";
            break;
        case QUEEN:
            return @"Queen";
            break;
    }
}

```

```

        case JACK:
            return @"Jack";
            break;
        case ACE:
            return @"Ace";
            break;
        default:
            return [NSString stringWithFormat:@"%i",value];
            break;
    }
}

- (NSString *)description{
    return [NSString stringWithFormat:@"The %@ of %@ with a UID of %i has a num value of %d",[MNCard faceValueToCardName:faceValue],suit,serialNumber,numberValue];
}

- (void)dealloc {
    //[frontImage release];
    [super dealloc];
}

@end
//
// MNDeck.m
// MNCards
//
// Created by Moo on 6/11/09.
// Copyright 2009 Micmoo. All rights reserved.
//

#import "MNDeck.h"
#import "MNCard.h"
#import "NSMutableArray+Shuffle.h"

@implementation MNDeck
@synthesize deck,size;

- (id)initWithNumberOfDecks:(int)numDecks andMode:(MNCardValueMode)modes{
    self = [super init];
    size = numDecks * 52; //# of cards = 52 * numDecks
    deck = [[MNDeck createDeck:numDecks andMode:0] retain];
    mode = modes;
    numberOfDecks = numDecks;
    return self;
}

- (NSMutableArray *)dealCards:(int)numCards { // deals n # of cards

```

```

NSMutableArray *array = [[NSMutableArray alloc] init];
for (int i = 1; i <= numCards; i++){ // if we don't have enough cards,
    //return error at end of available cards
    if (size < (numCards - i)){
        [array addObject:[NSNumber numberWithInt:NotEnoughCardsInDeck]];
        return array;
    }
    [array addObject:[deck lastObject]]; //add last card of array to array
    [deck removeObject:[deck lastObject]]; //remove from deck and return
    size--;
}
[array addObject:[NSNumber numberWithInt:EverythingisOK]];
return [array autorelease];
}

- (int)count{
    return size;
}

- (void)shuffle{
    [deck shuffle];
}

- (MNCard *)draw{ // draw last card
    if (size == 0) return nil;
    MNCard *card = [deck lastObject]; //??
    [deck removeObject:card];
    size--;
    return card;
}

+ (NSMutableArray *)createDeck:(int)numDecks andMode:(MNCardValueMode)modes{
    // Class Method that creates deck of MNCards ... factory of sorts.
    NSMutableArray *deck = [[NSMutableArray alloc] init];
    int uid = 0; //Triple for loop...
    for(int k = 0; k < numDecks; k++){ //loop as many times as decks that were passed
        for (int i = 0; i < 4; i++){ // Loop per suit
            for (int j = 1; j <= 13; j++){ //Ace - King
                NSString *suit;
                NSString *string;
                switch(i){
                    case 0: //If its the first time through, do spades
                        suit = SPADES;
                        string = @"s";
                        break;
                    case 1:
                        suit = CLUBS;
                        string = @"c";
                        break;
                }
            }
        }
    }
}

```

```

        case 2:
            string = @"h";
            suit = HEARTS;
            break;
        case 3:
            string = @"d";
            suit = DIAMONDS;
            break;
    }
    int value;

    if (modez == RummyMode && j > 10 && j < 14) value = 10; //Not
implemented
    else value = j;
    //This all loads the card image
    NSString *imageLocation = [[NSString stringWithFormat:@"%c%d.png",
        lowercaseString];
        [suit characterAtIndex:0], j]

        UIImage *image = [UIImage imageNamed:imageLocation];
        MNCard *card = [[MNCard alloc] initWithId:uid //Makse a card
            suit:suit
            faceValue:j
            frontImage:image
            andNumberValue:value];

        [deck addObject:card];
        uid++; //Adds it to deck array
        [image release];
    }
}
return [deck autorelease];
}

- (void)dealloc{
    for(int i = 0; i < size; i++){
        [[deck objectAtIndex:i] release];
    }
    [deck release];
    [super dealloc];
}

@end
//
// NSMutableArray+Shuffle.m
// MNCards
//
// Created by Moo on 6/12/09.

```

```
// Copyright 2009 Micmoo. All rights reserved.
//

#import "NSMutableArray+Shuffle.h"

@implementation NSMutableArray (Shuffling)

- (void)shuffle
{
    NSUInteger count = [self count];
    for (NSUInteger i = 0; i < count; ++i) {
        // Select a random element between i and end of array to swap with.
        int nElements = count - i;
        int n = (arc4random() % nElements) + i;
        [self exchangeObjectAtIndex:i withObjectAtIndex:n];
    }
}

@end

//
// main.m
// Run 21
//
// Created by Moo on 6/13/09.
// Copyright Micmoo 2009. All rights reserved.
//

#import <UIKit/UIKit.h>

int main(int argc, char *argv[]) {

    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];
    int retVal = UIApplicationMain(argc, argv, nil, nil);
    [pool release];
    return retVal;
}
```