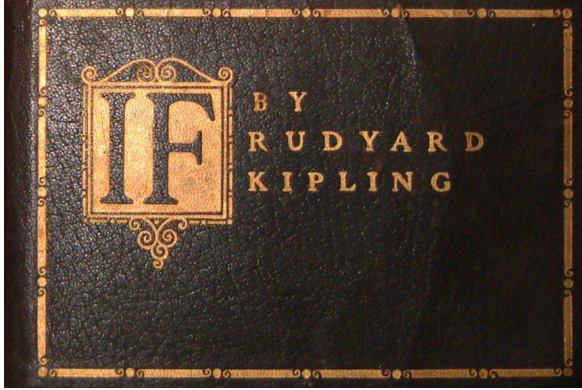
Conditionals

Conditional Pseudocode



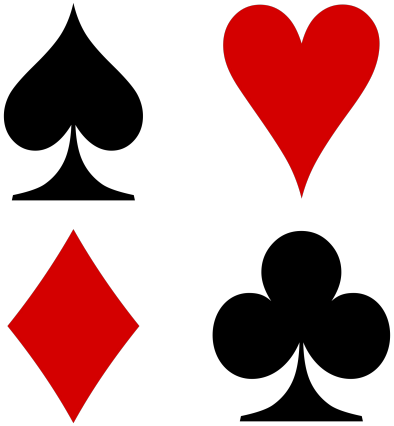
If <statement>

← *Tab* → /* Code that occurs if the statement is true */

Else

← *Tab* → /*Code that occurs if the statement is false */

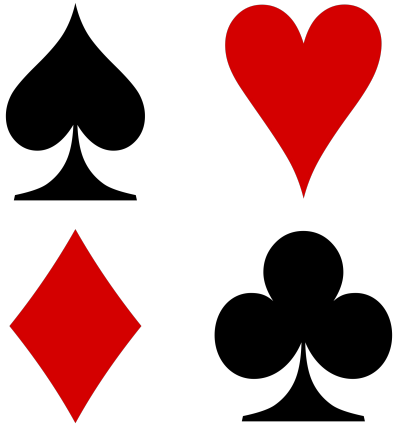
Conditional Pseudocode: Card Game V1



Pseudocode Review

1. Identify inputs, outputs, and assumptions
2. Define and set variables (denoted by < >)
3. Write general code (can be understood by a human, but can be easily translated into C code)

Conditional Pseudocode: Card Game V1

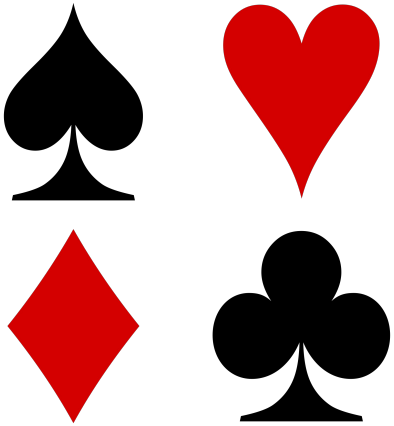


Inputs: One variable will be defined as red (r) or black (b)

Outputs: The scores of each team

Assumptions: The cards used will only be red or black

Conditional Pseudocode - Card Game V1



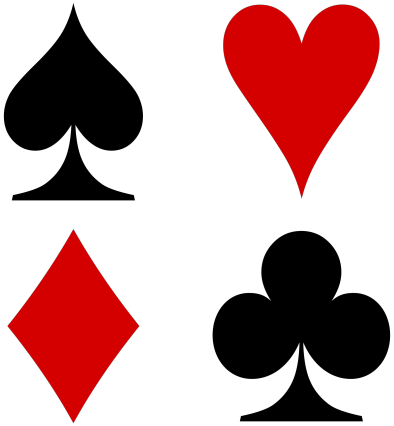
Variable Initialization

<card> = ‘r’

<window_score> = 0

<hallway_score> = 0

Conditional Pseudocode - Card Game V1



If statement

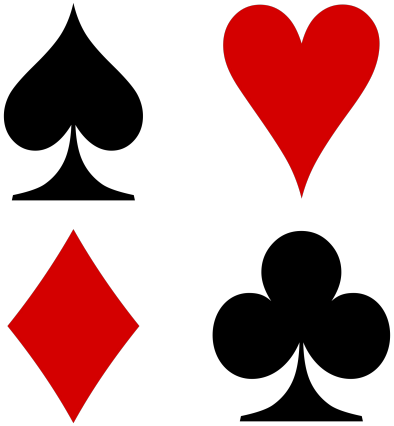
```
If card equals ‘r’

    <window_score> = <window_score + 1

Else

    <hallway_score> = <hallway_score> + 1
```

Conditional Pseudocode - Card Game V1

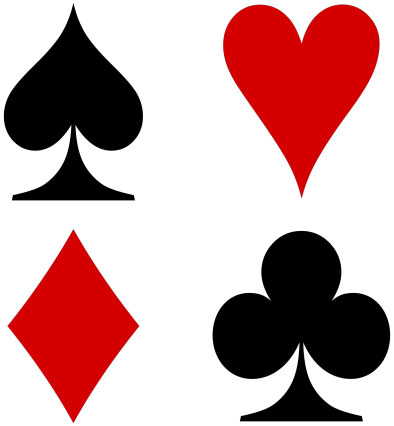


Outputs

Display “The window score is <window_score>”

Display “The hallway score is <hallway_score>”

Conditional Pseudocode - Card Game V1



```
<card> = 'r'

<window_score> = 0

<hallway_score> = 0

If card equals 'r'

        <window_score> = <window_score + 1

Else

        <hallway_score> = <hallway_score> + 1

Display "The window score is <window_score>"

Display "The hallway score is <hallway_score>"
```

Relational Operators

“Equals” → ==

“Does not equal” → !=

“Less than” → <

“Greater than” → >

“Greater than or equal to” → >=

“Less than or equal to” → <=

If/Else statement coding - C

All code within the if statement is indented and within curly braces

```
If (statement) {

    /*Code goes here*/

} else {

    /*More code goes here*/

}
```

The else statement (if present) is joined to the if statement

SCANNING IN VARIABLES

Use *scanf()* to scan in a variable from the user

Three parts:

1. scanf → scans a variable from the user
2. (<variable sign>, → takes in a specific type of variable (int: %d; Float: %f; char: %c)
3. &<variable_name>); → The name of the variable being scanned, WITH AN AMPERSAND IN FRONT

SCANNING IN VARIABLES

```
#include <stdio.h>

int main(void) {

        int month;                          /*Initalizes a variable*/

        scanf("%d", &month);                /*Scans in a number from the
                                            user*/

        printf("The month is: %d\n",
month);                                     /*Prints out that number*/A

        return 0;

}
```

scanf/if Practice

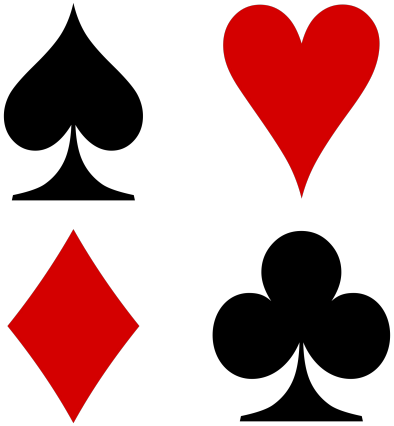
2 Parts:

1. Scan a variable into a variable named <month>, and print it out

3. Value-check <month> using if statements
   a. <month> only be between 1-12 (inclusive)
   b. If it is, print out the number of the month
   c. If it is not, print out the error message, “This month does not exist”

4. Write this in a file called *month.c* and submit to Github

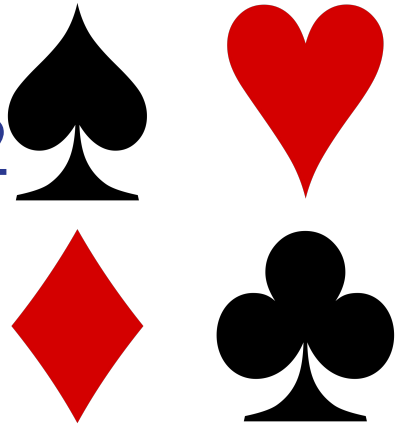
Card Game V2



Hallway side gets a point if card value is greater than 10, or if card value is less than 4

Window side gets a point otherwise

Conditional Pseudocode - Card Game V2



If/Else if statement

```
If card greater than 10

    <hallway_score> = <hallway_score + 1

Else if card less than 4

    <hallway_score> = <hallway_score> + 1

Else

    <window_score> = <window_score> + 1
```

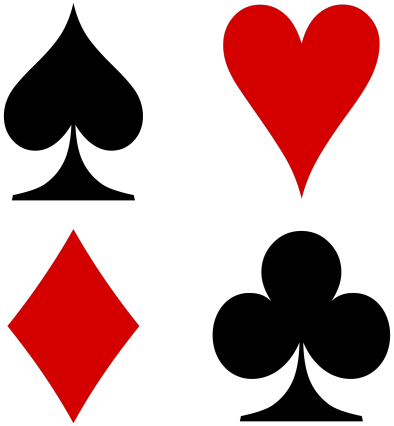
If/Else If/Else statement coding - C

```
If <statement> {
    /*Code goes here*/
} else if <statement>{
    /*More code goes here*/
} else {
    /*More code*/
}
```

Card Game V3

- If card is black, and is greater than 8 → hallway side +1
- If card is black, and is less than or equal to 8 → window side +1
- If card is red, and is greater than 8 → window side +1
- If card is red, and is less than or equal to 8 → hallway side +1

Card Game V3



- Two ways to solve this problem…
  - 1) Nested if statements
  - 2) Logical Operators

Nested If Statements

- *If* statements can be placed inside other if statements

```
if (statement) {

    if (statement) {
        /*Code goes here*/
    } else {
        /*More code here*/
    }

}
```

Nested If Statements

- *If* statements can also be placed inside *else* statements

```
if (statement) {
    /* Code here (may include an if statement */
} else {

    if (statement) {
        /*Code goes here*/
    } else {
        /*More code here*/
    }
}
```

Logical Operators

- Allow conditional statements to be linked together
- Three kinds
  - && → And
  - || → Or
  - ! → Not (negates the statement)

Logical Operators

- Three kinds
  - && → And
  - || → Or
  - ! → Not (negates the statement)

```
/* If card is black, and greater than 8 */

If (card == 'b' && card >= 8) {
    /* Code */
}
```

&& Logical Operator

- Both parts have to be *true* for the entire statement to be *true*

Date == 10 && Month == “october” ⇒ *True*

True *True*

Date == 10 && Month == “june” ⇒ *False*

True *False*

&& Logical Operator: Truth Table

- Formal notation

	Statement 1 (**P**)	Statement 2 (**Q**)	Final Value (**P && Q**)
&&	T	T	T
&&	T	F	F
&&	F	T	F
&&	F	F	F

|| Logical Operator

- Only one of the statements have to be *true* for the entire statement to be *true*

Obama == “Prez” || Romney == “Prez” ⇒ *True*

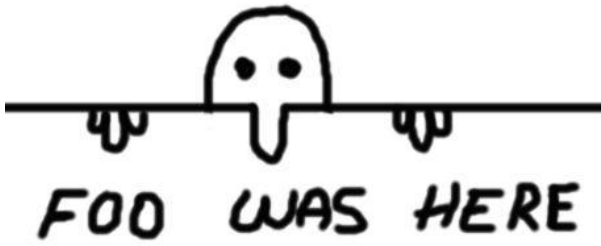
True *False*

|| Logical Operator: Truth Table

- Formal notation

	Statement 1 (**P**)	Statement 2 (**Q**)	Final Value (**P && Q**)
||	T	T	T
||	T	F	T
||	F	T	T
||	F	F	F

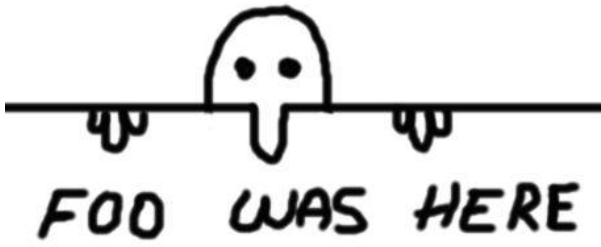
! Logical Operator



- Negates the statement
- Commonly used in mathematical expressions

```
int foo = 5;
if (foo != 6) {                    /*If foo does not equal 6*/
    printf("Hello World\n");
}
```

! Logical Operator



- Can also be used in front of an entire statement

```
int foo = 5;
if !(foo != 6) {        /*Makes the statement the negation (opposite) of its original value */

    printf("Hello World\n");
}
```