

# Introduction to Computing – B142L

<http://cnfolio.com/IndexIntroToComputing>

1. **Getting started with C programming**
2. **Data types**
3. **Numbers**
4. **Control structures**
5. **Standard input and output**
6. **Arrays**
7. **Functions**
8. **Memory allocation**
9. **Preprocessor directives**
10. **Enums and structs**

*October 26, 2009*

*November 2, 2009*

# Obtain input

① Declare variable

```
int card;
```

② Use `scanf()` function

```
scanf( "%d", &card );
```

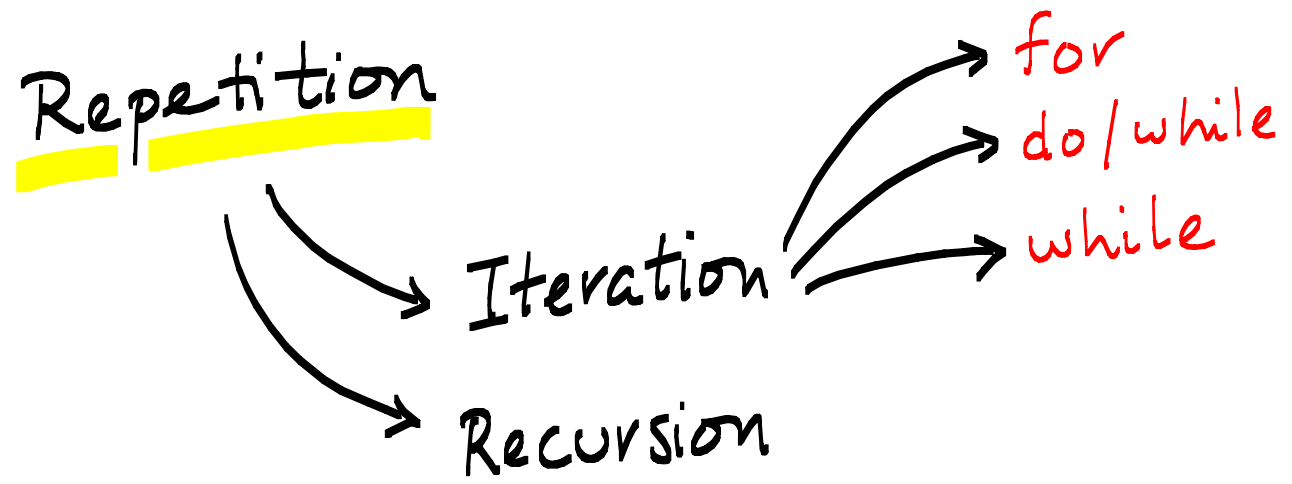
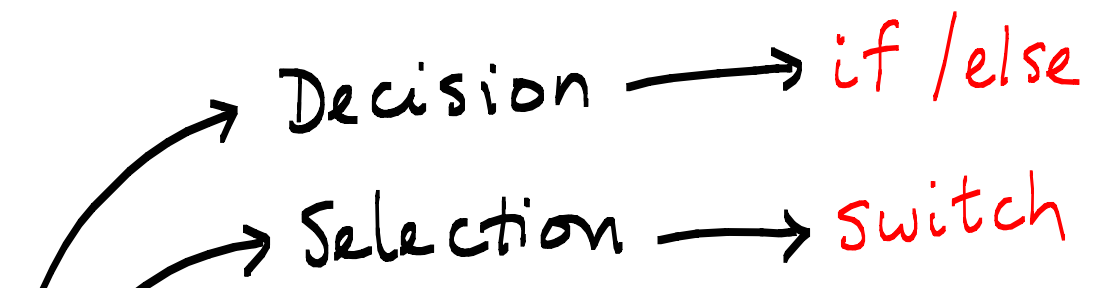
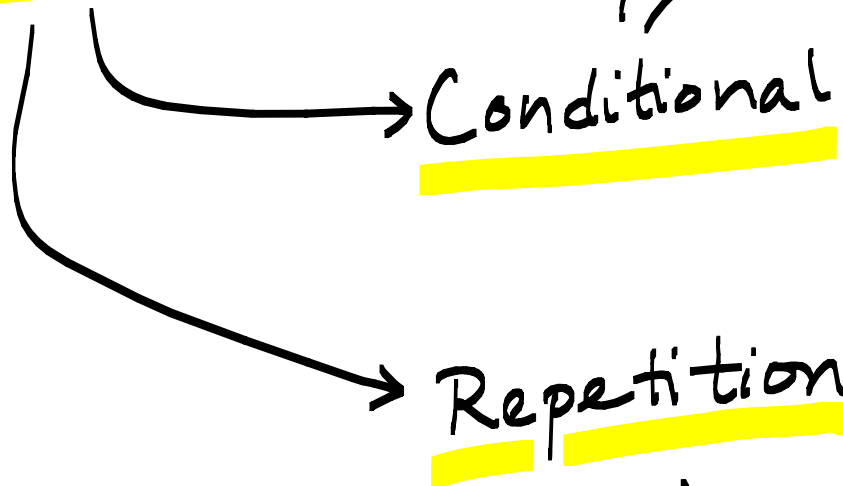
variable name

integer format

address operator

```
1 | int main( void )
2 | {
3 |     int card;
4 |
5 |     scanf( "%d", &card );
6 |     printf( "You entered number %d", card );
7 | }
```

Control structures



All programs can be written using

- sequential

- conditional

- repetition

} instructions

Paper written  
by Bohm & Jacopini  
in 1966

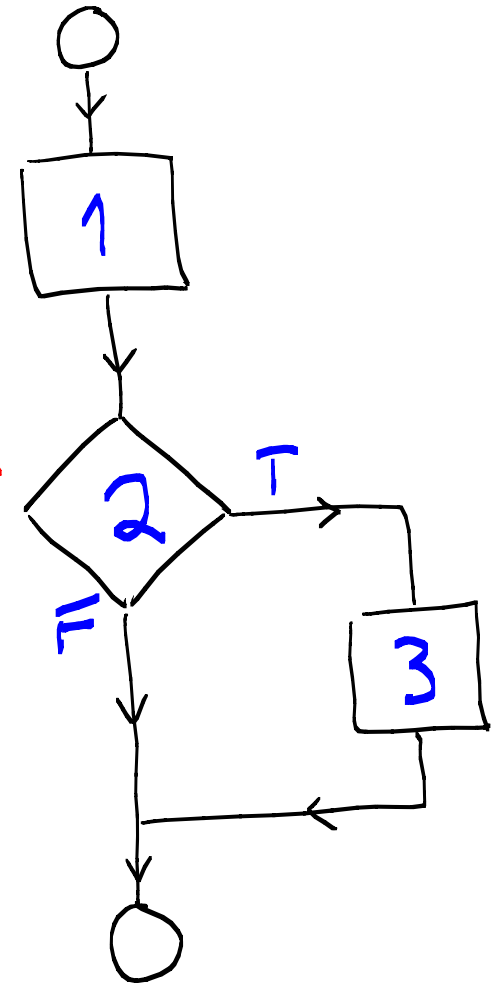
# Single decision

- ① display have a nice day
- ② if raining
- ③ display reminder to bring umbrella

pseudocode

test condition

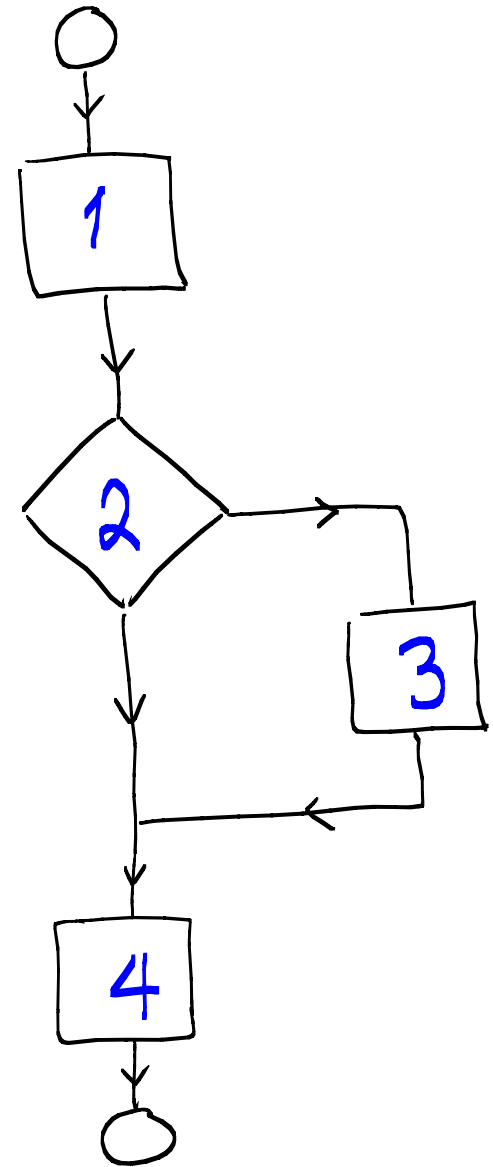
D structures



```
1 | int main( void )
2 | {
3 |     int raining = 1;
4 |
5 |     printf( "Have a nice day \n" );
6 |
7 |     if ( raining )
8 |         printf( "Bring an umbrella" );
9 | }
```

# Single decision

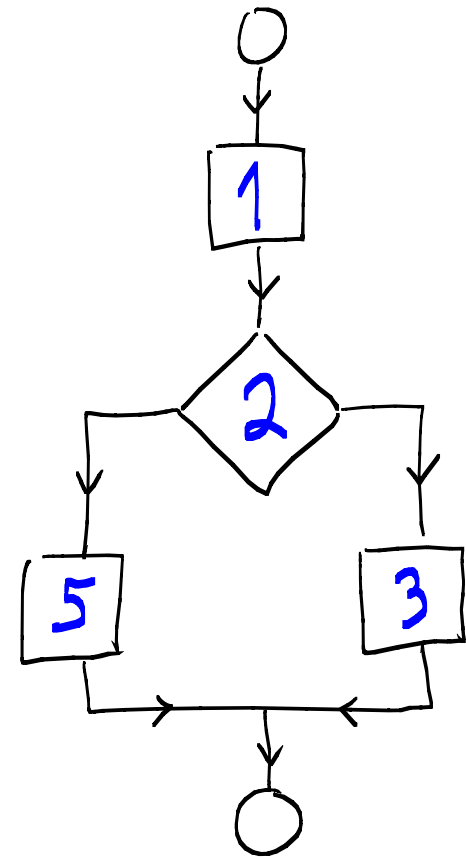
- ① obtain input number
- ② if multiple of 5
- ③ display message that multiple found
- ④ display remainder of division by 5



```
1 | int main( void )
2 | {
3 |     int number;
4 |
5 |     scanf( "%d", &number );
6 |
7 |     if ( ( number % 5 ) == 0 )
8 |     {
9 |         printf( "Number is multiple of 5 \n" );
10 |        printf( "Nice work \n" );
11 |    }
12 |
13 |    printf( "The remainder is %d", number % 5 );
14 | }
```

# if/else decisions

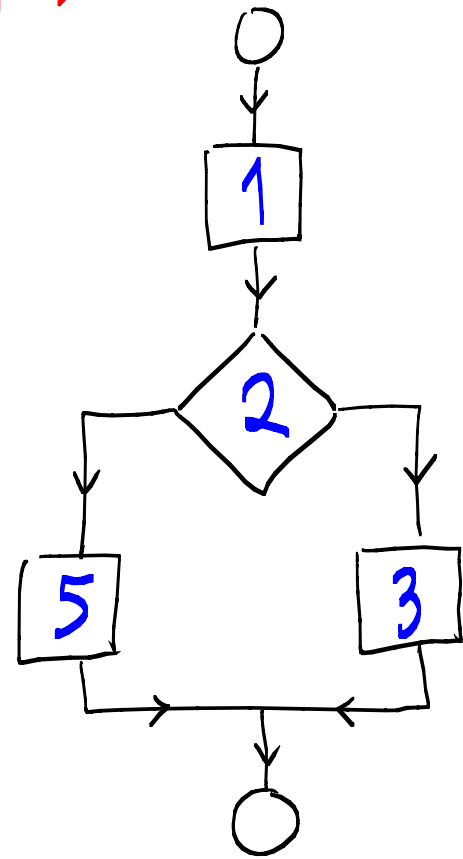
- ① obtain 2 cards
- ② if sum of cards equals 21
- ③ display blackjack message
- ④ else
- ⑤ display sum of cards



```
1 int main( void )
2 {
3     int card1, card2, sum;
4
5     scanf( "%d %d", &card1, &card2 );
6     sum = card1 + card2;
7
8     if ( sum == 21 )
9     {
10         printf( "Blackjack! " );
11         printf( "Congratulations" );
12     }
13     else
14     {
15         printf( "Sum is %d, ", sum );
16         printf( "Try again" );
17     }
18 }
```

# if/else decisions (with negative logic)

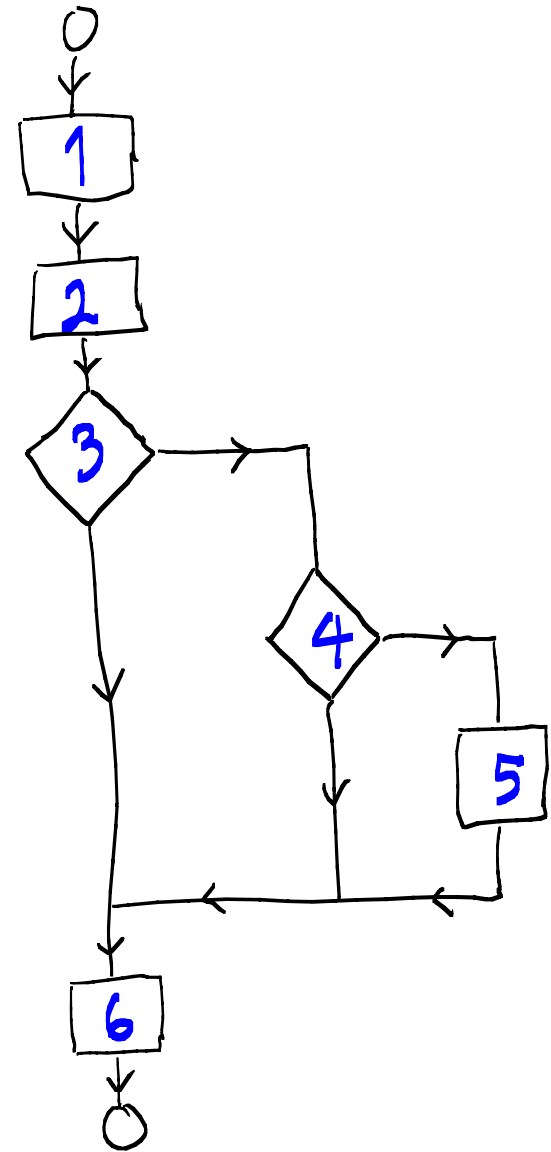
- ① obtain 2 cards
- ② if sum of cards not equal to 21
- ③ display sum of cards
- ④ else
- ⑤ display blackjack message



```
1 int main( void )
2 {
3     int card1, card2, sum;
4
5     scanf( "%d %d", &card1, &card2 );
6     sum = card1 + card2;
7
8     if ( sum != 21 )
9     {
10        printf( "Sum is %d, ", sum );
11        printf( "Try again" );
12    }
13    else
14    {
15        printf( "Blackjack! " );
16        printf( "Congratulations" );
17    }
18 }
```

# Nested decisions

- ① obtain 2 cards
- ② calculate sum of cards
- ③ if sum of cards less than 12
- ④ if any card is an ace
- ⑤ add 10 to sum
- ⑥ display sum

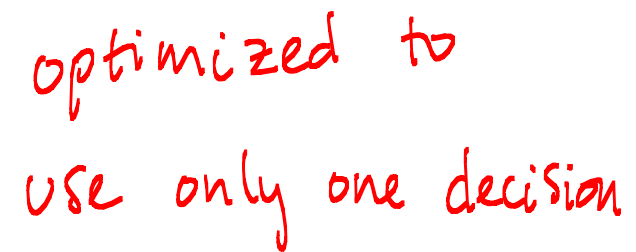


```
1 int main( void )
2 {
3     int card1, card2, sum;
4
5     scanf( "%d %d", &card1, &card2 );
6     sum = card1 + card2;
7
8     if ( sum < 12 )
9     {
10         if ( ( card1 == 1 ) || ( card2 == 1 ) )
11             sum = sum + 10;
12     }
13
14     printf( "Sum of cards is %d", sum );
15 }
```

*nested decisions*



*optimized to use only one decision*

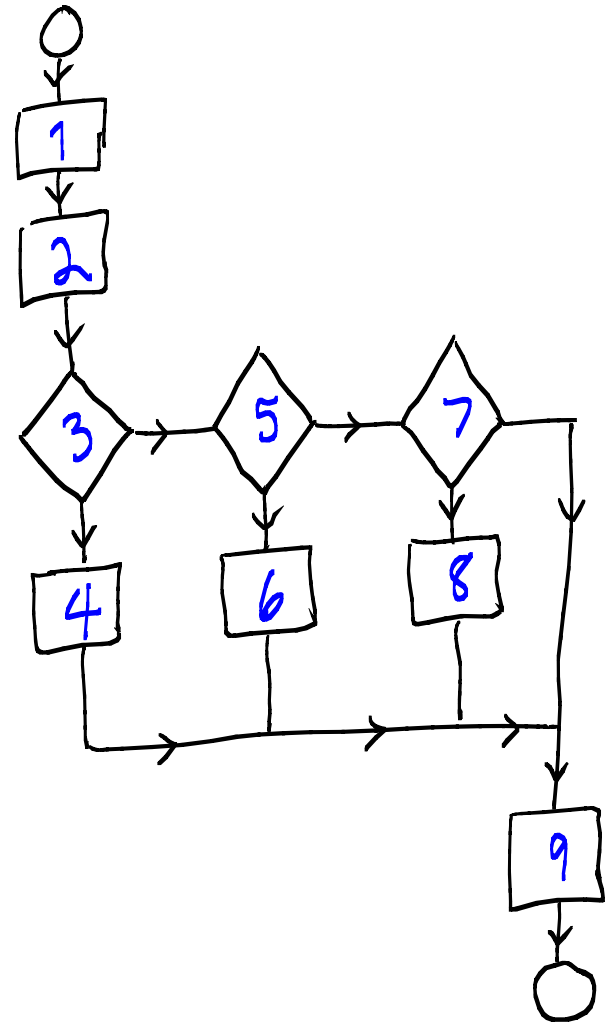


---

```
1 int main( void )
2 {
3     int card1, card2, sum;
4
5     scanf( "%d %d", &card1, &card2 );
6     sum = card1 + card2;
7
8     if ( ( sum < 12 ) && ( ( card1 == 1 ) || ( card2 == 1 ) ) )
9         sum = sum + 10;
10
11     printf( "Sum of cards is %d", sum );
12 }
```

# Chained decisions

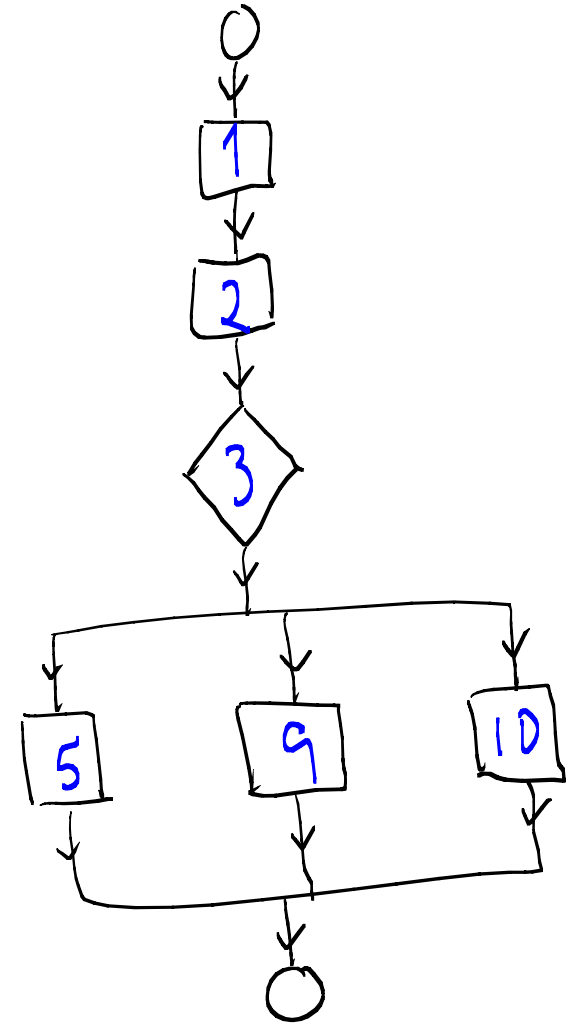
- ① obtain 2 cards
- ② calculate sum of cards
- ③ if sum equal to 21
- ④ display blackjack message
- ⑤ else if 2 cards is actually a pair
- ⑥ suggest that player split the pair
- ⑦ else if sum is more than 17
- ⑧ suggest that player stand the hand
- ⑨ display the sum



```
1 #include <stdio.h>
2
3 int main( void )
4 {
5     int card1, card2, sum;
6
7     scanf( "%d %d", &card1, &card2 );
8
9     sum = card1 + card2;
10
11     if ( sum == 21 )
12         printf( "Blackjack!" );
13     else if ( card1 == card2 )
14         printf( "Split your cards!" );
15     else if ( sum > 17 )
16         printf( "Stand!" );
17
18     printf( "\nSum of cards is %d", sum );
19 }
```

# Selection decisions

- ① obtain 2 cards
- ② calculate sum of cards
- ③ evaluate value of sum:
- ④ 21
- ⑤ display blackjack message
- ⑥ 20
- ⑦ 19
- ⑧ 18
- ⑨ display stand message
- ⑩ default
- ⑪ display sum

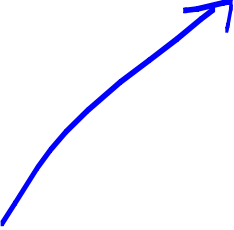


```
1 #include <stdio.h>
2
3 int main( void )
4 {
5     int card1, card2, sum;
6
7     scanf( "%d %d", &card1, &card2 );
8
9     sum = card1 + card2;
10
11     switch ( sum )
12     {
13         case 21 :
14             printf( "Blackjack!" );
15         case 20 :
16         case 19 :
17         case 18 :
18             printf( "Stand!" );
19         default :
20             printf( "\nSum of cards is %d", sum );
21     }
22 }
```

Chained decisions are single path!

Selection decisions could be multiple paths!

switch  
statement  
requires  
integer variable



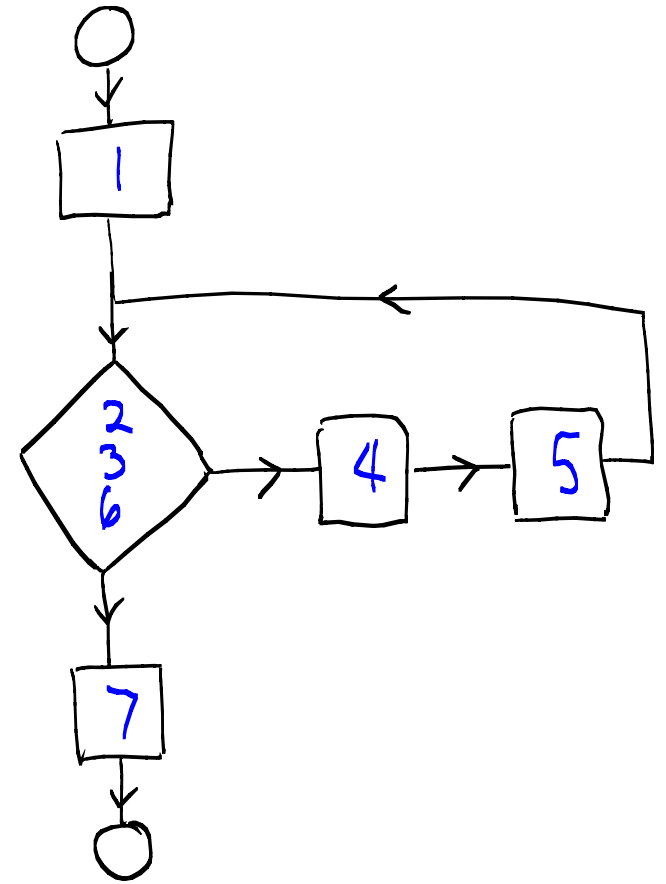
Be careful of division by zero  
in test conditions.

Use extra `printf()` statements  
to help trace program logic.

Be careful of compound statements  
which must be enclosed in a pair of `{ }`

# Iteration using for loops

- ① setup variables
- ② initialize counter ← counter controlled
- ③ evaluate test condition ← counter controlled
- ④ obtain input card value
- ⑤ add input to sum
- ⑥ update counter ← counter controlled
- ⑦ display sum of all cards



```
1 #include <stdio.h>
2
3 int main( void )
4 {
5     int card = 0;
6     int sum = 0;
7     int counter;
8
9     for ( counter = 0; counter < 3 ; counter++ )
10    {
11        scanf( "%d", &card );
12        sum = sum + card;
13    }
14
15    printf( "Sum of cards is %d", sum );
16 }
```

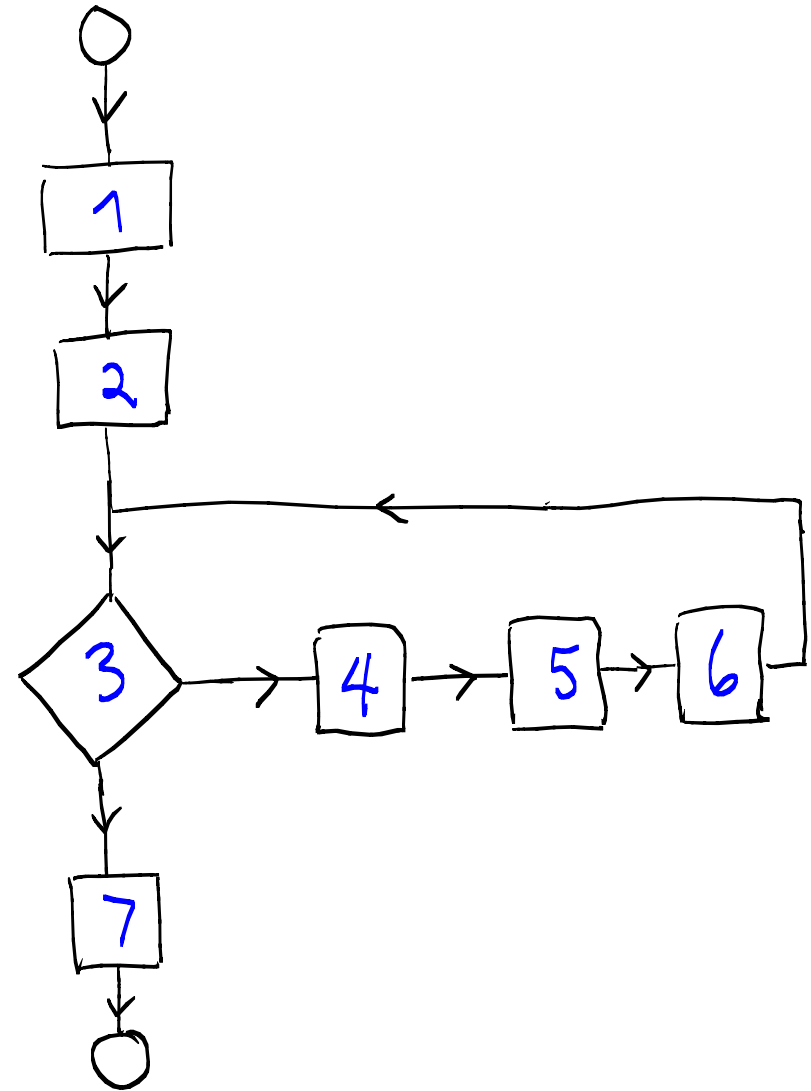
*initialization*

*test condition*

*increment step*

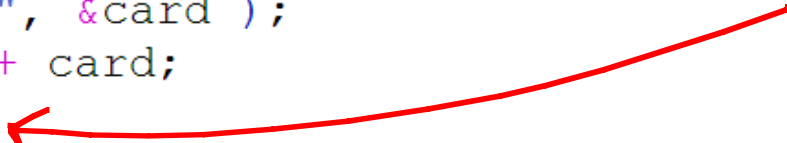
# Iteration using while loops

- ① setup variables
- ② initialize counter
- ③ evaluate test condition
- ④ obtain input card value
- ⑤ add input to sum
- ⑥ update counter
- ⑦ display sum of all cards



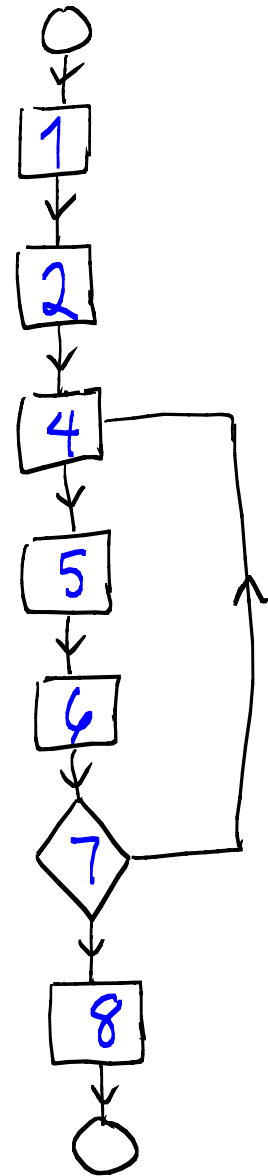
```
1 int main( void )
2 {
3     int card = 0;
4     int sum = 0;
5
6     int counter = 0;
7
8     while ( counter < 4 )
9     {
10         scanf( "%d", &card );
11         sum = sum + card;
12         counter++;
13     }
14
15     printf( "Sum is %d", sum );
16 }
```

remember to update  
the counter to avoid  
infinite loops



# Iteration using do/while loops

- ① setup variables
- ② initialize counter
- ③ start loop
- ④ obtain input card value
- ⑤ add input to sum
- ⑥ update counter
- ⑦ evaluate test condition
- ⑧ display sum of all cards



```

1 int main( void )
2 {
3     int card = 0;
4     int sum = 0;
5
6     int counter = 0;
7
8     do
9     {
10         scanf( "%d", &card );
11         sum = sum + card;
12         counter++;
13     }
14     while ( counter < 4 );
15
16     printf( "Sum is %d", sum );
17 }

```

## Dry run table

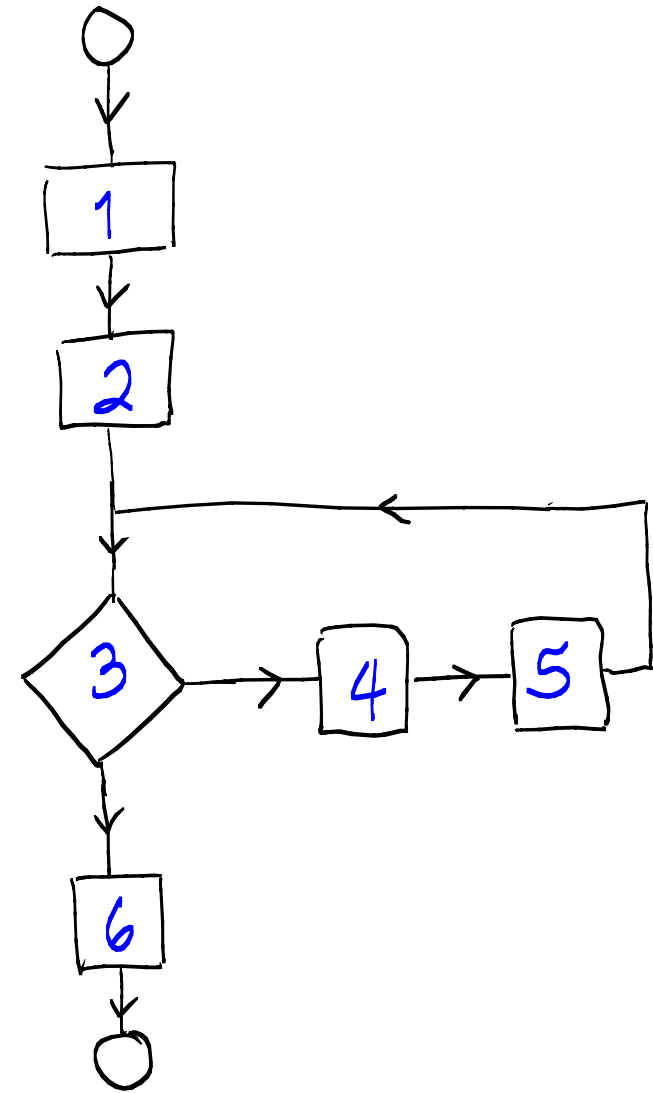
Step	card	sum	counter
1	0	0	0
2	2	2	1
3	3	5	2
4	5	10	3
5	6	16	4

# Iteration using while loops

- ① setup variables
- ② obtain input card value
- ③ compare input and sentinel value
- ④ add input to sum
- ⑤ obtain input card value
- ⑥ display sum of all cards

sentinel  
controlled

int  
data  
type



```
1 int main( void )
2 {
3     int card = 0;
4     int sum = 0;
5
6     while ( card != 15 )
7     {
8         scanf( "%d", &card );
9         sum = sum + card;
10    }
11
12    printf( "Sum is %d", sum );
13 }
```

Sentinel controlled loops are commonly associated with event based programming.

while (walking)  
    move forward by 1

while (eating)  
    increase health