1806ICT
Programming Fundamentals

Iteration

1

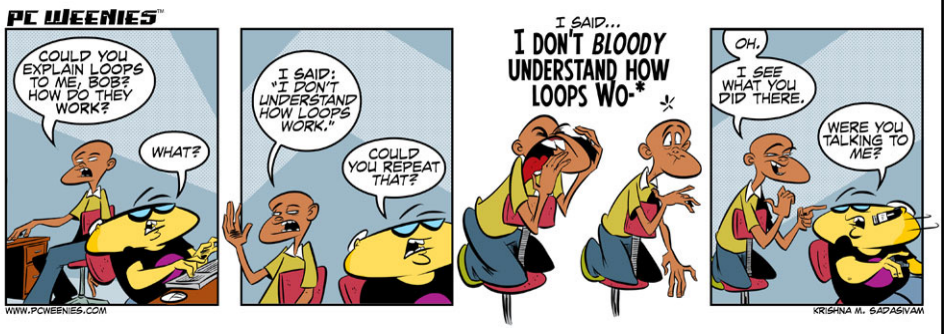
1

Topics

- **`while`** statement
- **`do-while`** statement
- **`for`** statement
- **`break`** statement
- Nested loops
- Infinite loops
- Examples

2

2



3

3

The **while** Statement

Implements the repetition in an algorithm

- Repeatedly executes a block of statements
- Tests a condition (Boolean expression) at the start of each iteration
- Terminates when condition becomes false (zero)

Syntax

```
while (Boolean_Expression)
{
        First_Statement
        Second_Statement
                ...
}
```

4

4

Example: **addnum.c**

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

5

5

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

Iteration Control

Initialize

Check condition

Update

6

6

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{

  return 0;
}
```

7

7

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;

  return 0;
}
```

only the variables sum and count are initialized to 0

8

8

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  return 0;
}
```

9

9

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {

  }

  return 0;
}
```

10

10

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }



  return 0;
}
```

11

11

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    co  ++;
```

Same as: `sum = sum + nextNum;`
Others: `-=`, `*=`, `/=`, etc.

12

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }
```

Same as: `count = count + 1;`
Decrement: `count --;`

13

Example: **addnum.c** (cont)

Read in numbers, add them, and print their sum and average

```
set sum to 0
set count to 0
input  totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);
  return 0;
}
```

14

14

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);
  return 0;
}
```

Example: **addnum.c** (cont)

totalNumbers	count	nextNum	sum
????	0	????	0.0

15

15

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);
  return 0;
}
```

Example: **addnum.c** (cont)

totalNumbers	count	nextNum	sum
????	0	????	0.0
3			

16

16

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);
  return 0;
}
```

Example: **addnum.c** (cont)

totalNumbers	count	nextNum	sum
????	0	????	0.0
3			
	1	4	4.0

17

17

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);
  return 0;
}
```

Example: **addnum.c** (cont)

totalNumbers	count	nextNum	sum
????	0	????	0.0
3			
	1	4	4.0
	2	-1	3.0

18

18

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count = 0, totalNumbers;
  scanf("%d", &totalNumbers);

  while (count < totalNumbers)
  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);
  return 0;
}
```

Example: **addnum.c** (cont)

totalNumbers	count	nextNum	sum
????	0	????	0.0
3			
	1	4	4.0
	2	-1	3.0
	3	6.3	9.3

19

19

Common Mistakes in **while** – *"one liners"*

```
while (num < minimum)
  scanf("%d", &num);
  printf("Number must be greater than %d.\n", minimum);
  printf("Please try again.\n");
```

```
while (num < minimum)
{
  scanf("%d", &num);
}

printf("Number must be greater than %d.\n", minimum);
printf("Please try again.\n");
```

20

20

Common Mistakes in **while** -- "*one liners*" (cont)

```
while (num < minimum)
  scanf("%d", &num);
  printf("Number must be greater than %d.\n", minimum);
  printf("Please try again.\n");
```

```
while (num < minimum)
{
  scanf("%d", &num);
  printf("Number must be greater than %d.\n", minimum);
  printf("Please try again.\n");
}
```

21

21

Common Mistakes in **while** -- extra semi-colon;

```
while (num < minimum);
{
  scanf("%d", &num);
  printf("Number must be greater than %d.\n", minimum);
  printf("Please try again.\n");
}
```

Marks the end of the while-block -- usual cause of infinite loops

22

22

`scanf()`

- `scanf()` returns the number of items that were successfully read and assigned to variables
- When a read value does not match the expected data type, `scanf()` stops reading and immediately returns
- In the case of an input failure before any data is read, or when input ends, a special value is returned: **EOF**

23

23

Checking for End-of-Input / End-of-File in `while`

Read in numbers, add them, and print their sum

```
set sum to 0

input nextNum
check if end of input
while  (not end of input)
{
   add nextNum to sum
   input nextNum
   check if end of input
}
etc...etc...etc...
```

24

24

Checking for End-of-Input / End-of-File in **while** (cont)

Read in numbers, add them, and print their sum

set sum to 0

input nextNum
check if end of input
while (not end of input)
{
add nextNum to sum
input nextNum
check if end of input
}
etc...etc...etc...

etc...etc...etc...

```
float nextNum;
float sum = 0.0;


while ( scanf("%f",&nextNum) > 0 )
{

  sum += nextNum;

}
```

etc...etc...etc...

25

25

Topics

✓ **while** statement

- **do-while** statement
- **for** statement
- **break** statement
- Nested loops
- Infinite loops
- Examples

26

26

The **do-while** Statement

- First, the loop body is executed.
- Then the Boolean expression is checked.
  - As long as it is true, the loop is executed again.
  - If it is false, the loop is exited.
- Similar to a **while** statement, except that the loop body is executed at least once

Syntax

```
do
{
    Body_Statements
}
while (Boolean_Expression);
```

27

27

Example: addDoWhile.c

Read in numbers, add them, and print their sum

set sum to 0

do
{
add nextNum to sum
input nextNum
check if end of input
} while (not end of input);
etc...etc...etc...

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum
\**********************************/
int main()
{
   float nextNum = 0.0;
   float sum = 0.0;
   do
   {
        sum += nextNum;
   } while (scanf("%f",&nextNum) > 0);

   printf("Sum was %f\n", sum);
   return 0;
}
```

28

28

Topics

- ✓ `while` statement
- ✓ `do-while` statement
- `for` statement
- `break` statement
- Nested loops
- Infinite loops
- Examples

29

29

The `for` Statement

- Form of loop which allows for initialization and iteration control
- Syntax:

```
for ( initialization; condition; update )
{
    instruction block
}
```

Careful! A semi-colon here marks the end of the instruction block!

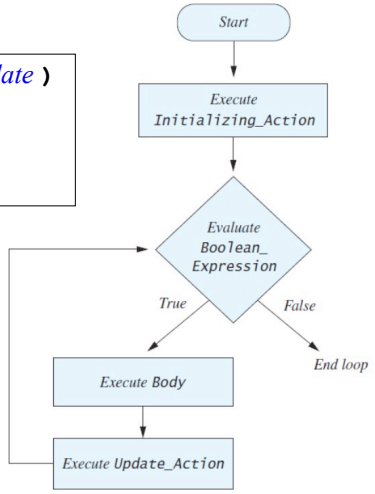
30

30

Semantics of the `for` Statement

```
for (Initializing_Action; Boolean_Expression; Update_Action)
    Body
```

```
for ( initialization ; condition ; update )
{
    instruction block
}
```



31

Example: addfor.c

Read in numbers, add them, and print the sum and the average

```
set sum to 0
set count to 0
input totalNumbers

while  (count < totalNumbers)
{
   input nextNum
   add nextNum to sum
   add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count
```

32

32

Example: **addfor.c** (cont)

Read in numbers, add them, and print the sum and the average

set sum to 0
set count to 0
input totalNumbers

while (count < totalNumbers)
{
input nextNum
add nextNum to sum
add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum = 0.0;
  int count, totalNumbers;

  scanf("%d", &totalNumbers);

  for ( count=0;
        count < totalNumbers;
        count++ )
  {
    scanf("%f", &nextNum);
    sum += nextNum;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);

  return 0;
}
```

33

33

Example: **addfor.c** (cont)

Read in numbers, add them, and print the sum and the average

set sum to 0
set count to 0

input totalNumbers

while (count < totalNumbers)
{
input nextNum
add nextNum to sum
add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  float nextNum, sum
  int count, totalNu

  scanf("%d", &total bers);

  for ( count=0;
        count < totalNumbers;
        count++ )
  {
    scanf("%f", &nextNum);
    sum += nextNum;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);

  return 0;
}
```

Initialize

34

34

Example: **addfor.c** (cont)

Read in numbers, add them, and print the sum and the average

set sum to 0
set count to 0

input totalNumbers

while (**count < totalNumbers**)
{
  input nextNum
  add nextNum to sum
  add 1 to count
}

output "Sum was" sum
output "Mean was" sum/count

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{
  flo
  int

  scanf("%d", &totalNu  s);

  for ( count=0;
        count < totalNumbers;
        count++ )
  {
    scanf("%f", &nextNum);
    sum += nextNum;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",sum/count);

  return 0;
}
```

Check condition

35

35

Example: **addfor.c** (cont)

Read in numbers, add them, and print the sum and the average

set sum to 0
set count to 0

input totalNumbers

while (count < totalNumbers)
{
  input n
  add n
  add 1
}

output "
output "

```
#include <stdio.h>
/**********************************\
 Read in numbers and add them up
 Print out the sum and the average
\**********************************/
int main()
{

  scan  &totalNumbers);

  for ( co  0;
        coun  < totalNumbers;
        count++ )

  &nextNum

  unt);

}
```

***Update* (aka *Increment Step*)**

IMPORTANT!!
The Update is performed AFTER the body of the loop

36

36

while and for

```c
#include <stdio.h>
int main()
{
  float nextNum, sum = 0.0;
  int count, totalNumbers;

  scanf("%d", &totalNumbers);

  count = 0;
  while (count < totalNumbers)

  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }
  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",
          sum/count);

  return 0;
}
```

```c
#include <stdio.h>
int main()
{
  float nextNum, sum = 0.0;
  int count, totalNumbers;

  scanf("%d", &totalNumbers);

  for ( count=0;
        count < totalNumbers;
        count++ )
  {
    scanf("%f", &nextNum);
    sum += nextNum;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",
          sum/count);

  return 0;
}
```

37

37

while and for (cont)

```c
#include <std
int main()
{
  float nextN
  int count, tota  s;

  scanf("%d"  totalNumbers);

  count = 0;
  while (count < totalNumbers)

  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",
          sum/count);

  return 0;
}
```

Initialize

```c
ude <stdio.h>
ain()

at nextNum, sum = 0.0;
int count, totalNumbers;

scanf("%d", &totalNumbers);

for ( count=0;
      count < totalNumbers;
      count++ )
{
  scanf("%f", &nextNum);
  sum += nextNum;
}

printf("Sum was %f\n",sum);
printf("Mean was %f\n",
        sum/count);

return 0;
}
```

38

38

while and for (cont)

```
#include <stdio.h>
int main()
{
  float nextNum, sum = 0.0;
  int count, totalNumbers;

  scanf("%d", &totalNumbers);

  count = 0;
  while (count < totalNumbers)

  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",
          sum/count);

  return 0;
}
```

Check condition

```
#include <stdio.h>
int main()
{
  float nextNum, sum = 0.0;
  int count, totalNumbers;

  scanf("%d", &totalNumbers);

  for ( count=0;
        count < totalNumbers;
        count++ )
  {
    scanf("%f", &nextNum);
    sum += nextNum;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",
          sum/count);

  return 0;
}
```

39

39

while and for (cont)

```
#include <stdio.h>
int main()
{
  float nextNum, sum = 0.0;
  int count, totalNumbers;

  scanf("%d", &totalNumbers);

  count = 0;
  while (count < totalNumbers)

  {
    scanf("%f", &nextNum);
    sum += nextNum;
    count++;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",
          sum/count);

  return 0;
}
```

Update

```
#include <stdio.h>
int main()
{
  float nextNum, sum = 0.0;
  int count, totalNumbers;

  scanf("%d", &totalNumbers);

  for ( count=0;
        count < totalNumbers;
        count++ )
  {
    scanf("%f", &nextNum);
    sum += nextNum;
  }

  printf("Sum was %f\n",sum);
  printf("Mean was %f\n",
          sum/count);

  return 0;
}
```

40

40

Topics

- ✓ `while` statement
- ✓ `do-while` statement
- ✓ `for` statement
- **`break`** statement
- Nested loops
- Infinite loops
- Examples

41

41

The **`break`** Statement

- Implements the "exit loop" primitive
- Causes flow of control to leave a loop block (**`while`** or **`for`**) immediately

42

42

Example: **recip.c**

Print out the reciprocals of numbers entered. Quit when 0 is entered

```
loop
{
  input nextNum
  if (nextNum is 0)
  {
    exit loop
  }
  else
  {
    output  1/nextNum
  }
}
```

43

43

Example: **recip.c** (cont)

Print out the reciprocals of numbers entered. Quit when 0 is entered

```
loop
{
  input nextNum
  if (nextNum is 0)
  {
    exit loop
  }
  else
  {
    output  1/nextNum
  }
}
```

```
#include <stdio.h>
/******************************\
  Print out the reciprocals of
  numbers entered. Quit when 0
  is entered
\******************************/

int main()
{
  float nextNum;

  return 0;
}
```

44

44

Example: **recip.c** (cont)

Print out the reciprocals of numbers entered. Quit when 0 is entered

```
loop
{
  input nextNum
  if (nextNum is 0)
  {
    exit loop
  }
  else
  {
    output  1/nextNum
  }
}
```

```
#include <stdio.h>
/*******************************\
  Print out the reciprocals of
  numbers entered. Quit when 0
  is entered
\*******************************/

int main()
{
  float nextNum;

  while (1)
  {

  }
  return 0;
}
```

"while (True)" infinite loop

45

45

Example: **recip.c** (cont)

Print out the reciprocals of numbers entered. Quit when 0 is entered

```
loop
{
  input nextNum
  if (nextNum is 0)
  {
    exit loop
  }
  else
  {
    output  1/nextNum
  }
}
```

```
#include <stdio.h>
/*******************************\
  Print out the reciprocals of
  numbers entered. Quit when 0
  is entered
\*******************************/

int main()
{
  float nextNum;

  while (1)
  {
    scanf("%f", &nextNum);

  }
  return 0;
}
```

46

46

Example: **recip.c** (cont)

Print out the reciprocals of numbers entered. Quit when 0 is entered

```
loop
{
  input nextNum
  if (nextNum is 0)
  {
    exit loop
  }
  else
  {
    output  1/nextNum
  }
}
```

```
#include <stdio.h>
/*******************************\
  Print out the reciprocals of
  numbers entered. Quit when 0
  is entered
\*******************************/

int main()
{
  float nextNum;

  while (1)
  {
    scanf("%f", &nextNum);
    if (nextNum == 0.0)
    {
      break;
    }
    else
    {
      printf("%f\n", 1/nextNum);
    }

  }
  return 0;
}
```

47

47

Example: **recip.c** (cont)

Print out the reciprocals of numbers entered. Quit when 0 is entered

```
loop
{
  input nextNum
  if (nextNum is 0)
  {
    exit loop
  }
  else
  {
    output  1/nextNum
  }
}
```

```
#include <stdio.h>
/*******************************\
  Print out the reciprocals of
  numbers entered. Quit when 0
  is entered
\*******************************/

int main()
{
  float nextNum;

  while (1)
  {
    scanf("%f", &nextNum);
    if (nextNum==0.0)
    {
      break;
    }
    else
    {
      printf("%f\n", 1/nextNum);
    }
  }

  return 0;
}
```

48

48

Example: **addpos.c**

Read in numbers, and add only the positive ones. Quit when input is 0

```
set sum to 0

loop
{
  input number
  if (number is zero)
  {
    exit loop
  }
  else if ( number is positive)
  {
    add number to sum
  }
}

output sum
```

49

49

Example: **addpos.c** (cont)

Read in numbers, and add only the positive ones. Quit when input is 0

```
set sum to 0

loop
{
  input number
  if (number is zero)
  {
    exit loop
  }
  else if ( number is positive)
  {
    add number to sum
  }
}

output sum
```

```
include <stdio.h>

/****************************
** Read in numbers, and add
** only the positive ones.
** Quit when input is 0
****************************/

int main()
{
  float num, sum = 0.0;



  printf("sum = %f\n", sum);
  return 0;
}
```

50

50

Example: **addpos.c** (cont)

Read in numbers, and add only the positive ones. Quit when input is 0

set sum to 0

loop
{
  input number
  if (number is zero)
  {
    exit loop
  }
  else if (number is positive)
  {
    add number to sum
  }

}

output sum

`scanf` returns EOF if an end of file occurs; otherwise it returns the number of items converted and assigned

```
include <stdio.h>

  float num = 0.0;

  while (scanf("%f", &num) > 0)
  {





    sum += num;
  }

  printf("sum = %f\n", sum);
  return 0;
}
```

51

51

Example: **addpos.c** (cont)

Read in numbers, and add only the positive ones. Quit when input is 0

set sum to 0

loop
{
  input number
  if (number is zero)
  {
    exit loop
  }
  else if (number is positive)
  {
    add number to sum
  }
}

output sum

```
include <stdio.h>

/****************************
** Read in numbers, and add
** only the positive ones.
** Quit when input is 0
****************************/

int main()
{
  float num, sum = 0.0;

  while (scanf("%f", &num) > 0)
  {
    if (num == 0)
      break;

    else if (num > 0)
      sum += num;
  }


  printf("sum = %f\n", sum);
  return 0;
}
```

52

52

Example: **addpos.c** (cont)

Read in numbers, and add only the positive ones. Quit when input is 0

```
set sum to 0

loop
{
  input number
  if (number is zero)
  {
     exit loop
  }
  else if ( number is positive)
  {
     add number to sum
  }
}

output sum
```

```
include <stdio.h>

/****************************
** Read in numbers, and add
** only the positive ones.
** Quit when input is 0
****************************/

int main()
{
  float num, sum = 0.0;

  while (scanf("%f", &num) > 0)
  {
    if (num == 0)
      break;


    else if (num > 0)
      sum += num;
  }


  printf("sum = %f\n", sum);
  return 0;
}
```

53

53

Example: **addpos.c** (cont)

These comparisons are ok despite `num` being of type `float`

```
include <stdio.h>

/****************************
** Read in numbers, and add
** only the positive ones.
** Quit when input is 0
****************************/

int main()
{
  float num, sum = 0.0;

  while (scanf("%f", &num) > 0)
  {
    if (num == 0)
      break;

    else if (num > 0)
      sum += num;
  }

  printf("sum = %f\n", sum);
  return 0;
}
```

54

54

Topics

- ✓ `while` statement
- ✓ `do-while` statement
- ✓ `for` statement
- ✓ `break` statement
- Nested loops
- Infinite loops
- Examples

55

55

Nested Loops

- Loops can be placed inside other loops
- The **`break`** statement applies to the innermost enclosing **`while`** or **`for`** statement

56

56

Example: **rect.c**

Print an m by n rectangle of asterisks

input width and height

for each row
{
for each column in the current row
{
print an asterisk
}
start next row
}

57

57

Example: **rect.c** (cont)

Print an m by n rectangle of asterisks

input width and height

for **each row**
{
for **each column** in the current row
{
print an asterisk
}
start next row
}

```
#include <stdio.h>

 /*  Print an m-by-n rectangle of
    asterisks */

int main()
{
  int rowc, colc, numrow, numcol;

  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);









  return 0;
}
```

58

58

Example: **rect.c** (cont)

Print an m by n rectangle of asterisks

input width and height

for each row
{
  for each column in the current row
  {
    print an asterisk
  }
  start next row
}

```
#include <stdio.h>

/*  Print an m-by-n rectangle of
    asterisks */

int main()
{
  int rowc, colc, numrow, numcol;

  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);

  for (rowc=0; rowc < numrow; rowc++)
  {









  }
  return 0;
}
```

59

59

Example: **rect.c** (cont)

Print an m by n rectangle of asterisks

input width and height

for each row
{
  for each column in the current row
  {
    print an asterisk
  }
  start next row
}

```
#include <stdio.h>

/*  Print an m-by-n rectangle of
    asterisks */

int main()
{
  int rowc, colc, numrow, numcol;

  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);

  for (rowc=0; rowc < numrow; rowc++)
  {
    for (colc=0; colc < numcol; colc++)
    {


    }

  }
  return 0;
}
```

60

60

Example: **rect.c** (cont)

Print an m by n rectangle of asterisks

input width and height

for each row
{
  for each column in the current row
  {
    print an asterisk
  }
  start next row
}

```
#include <stdio.h>

/*  Print an m-by-n rectangle of
    asterisks */

int main()
{
  int rowc, colc, numrow, numcol;

  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);

  for (rowc=0; rowc < numrow; rowc++)
  {
    for (colc=0; colc < numcol; colc++)

    {
      printf("*");
    }

  }
  return 0;
}
```

61

61

Example: **rect.c** (cont)

Print an m by n rectangle of asterisks

input width and height

for each row
{
  for each column in the current row
  {
    print an asterisk
  }
  start next row
}

```
#include <stdio.h>

/*  Print an m-by-n rectangle of
    asterisks */

int main()
{
  int rowc, colc, numrow, numcol;

  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);

  for (rowc=0; rowc < numrow; rowc++)
  {
    for (colc=0; colc < numcol; colc++)

    {
      printf("*");
    }

    printf("\n");
  }
  return 0;
}
```

62

62

Variation: **rect2.c**

Print an m by n rectangle of asterisks

input width and height

for each row
{
for each column in the current row
{
print an asterisk
}
start next row
}

```
#include <stdio.h>
/*  Print an m-by-n rectangle of
    asterisks */
int main()
{
  int rowc, colc, numrow, numcol;
  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);

  rowc = 0;
  while (rowc < numrow)
  {
    for (colc=0; colc < numcol; colc++)

    {
      printf("*");
    }

    printf("\n");
    rowc++;
  }
return 0;
}
```

63

63

Variation: **rect3.c**

Print an m by n rectangle of asterisks

input width and height

for each row
{
for each column in the current row
{
print an asterisk
}

start next row
}

```
#include <stdio.h>
/* Print an m-by-n rectangle of
   asterisks */
int main()
{
  int rowc, colc, numrow, numcol;

  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);

for (rowc=0; rowc < numrow; rowc++)
  {
    colc = 0;
    while (1)
    {
      printf("*");
      colc++;
      if (colc == numcol)
       { break; }
    }
    printf("\n");
  }
  return 0;
}
```

64

64

Variation: **rect3.c (cont)**

Print an m by n rectangle of asterisks

input width and height

for each row
{

The innermost enclosing loop for this break is the while-loop

start next row
}

```
#include <stdio.h>
/* Print an m-by-n rectangle of
   asterisks */
int main()
{
  int rowc, colc, numrow, numcol;

  printf("\nEnter width: ");
  scanf("%d", &numcol);
  printf("\nEnter height: ");
  scanf("%d", &numrow);

  for (rowc=0; rowc < numrow; rowc++)
  {
    colc = 0;
    while (1)
    {
      printf("*");
      colc++;
      if (colc == numcol)
        { break; }
    }
    printf("\n");
  }
  return 0;
}
```

65

65

Topics

- ✓ `while` statement
- ✓ `do-while` statement
- ✓ `for` statement
- ✓ `break` statement
- ✓ Nested loops
- Infinite loops
- Examples

66

66

Infinite Loops

```
while ( 1 )
{
  ...etc...etc...etc...
}
```

```
for ( ; 1 ; )
{
  ...etc...etc...etc...
}
```

```
for ( ; ; )
{
  ...etc...etc...etc...
}
```

67

67

Infinite Loops

```
while ( 1 )
{
  ...etc...etc...etc...
}
```

```
for ( ; 1 ; )
{
  ...etc...etc...etc...
}
```

```
for ( ; ; )
{
  ...etc...etc...etc...
}
```

Use an:

```
if ( condition )
{
  break;
}
```

statement to break the loop

68

68

Example: `asciiCheck.c`

```
while (1)
{
  printf("Enter bounds (low high): ");
  scanf("%d %d", &low, &high);

  if ((low >= 0) && (high <= 127) && (low < high))
  {
    break;
  }
  else
  {
    printf("Bad bounds. Try again.\n");
  }
}
```

69

69

Topics

- ✓ `while` statement
- ✓ `do-while` statement
- ✓ `for` statement
- ✓ `break` statement
- ✓ Nested loops
- ✓ Infinite loops
- Examples

70

70

`while` and `for`

A `for` loop can always be rewritten as an equivalent `while` loop, and vice-versa

71

71

Example: `asciiPrint`

Print out a section of the ASCII table

for each character from the lower bound to higher bound
{
  print its ascii value and ascii character
}

```
for ( ch = low; ch <= high; ch++ )
{
  printf("%d: %c\n", ch, ch);
}
```

asciiPrint1.

```
ch = low;
while ( ch <= high )
{
  printf("%d: %c\n", ch, ch);
  ch++;
}
```

asciiPrint2.

72

72

Example: ascii1.c

```
#include <stdio.h>

/* Print a section of
   the ASCII table */

#define  MIN   0
#define  MAX   127

int main()
{
  int  low, high;
  char ch;
```

```
  while (1)
  {
    printf("Enter bounds (low high): ");
    scanf("%d %d", &low, &high);

    if ((low >= MIN) && (high <= MAX)
       && (low < high))
    {
      break;
    }
    else
    {
      printf("Bad bounds. Retry.\n");
    }
  }

  for (ch=low; ch <= high; ch++)
  {
    printf("%d: %c\n", ch, ch);
  }

  return 0;
}
```

73

73

Example: ascii1.c (cont)

```
#include <stdio.h>

/* Print a section of
   the ASCII table */

#define  MIN   0
#define  MAX   127

int main()
{
  int  low, high;
  char ch;
```

```
  while (1)
  {
    printf("Enter bounds (low high):");
    scanf("%d %d", &low, &high);

    if ((low >= MIN) && (high <= MAX)
       && (low < high))
    {
      break;
    }
    else
    {
      printf("Bad bounds. Retry.\n");
    }
  }
```

Macro definition:

`#define identifier tokens`

All subsequent instances of *identifier* are replaced with its *token*

74

Example: CountConsonantsAndVowels

- Count the number of consonants and the number of vowels
- Non-alphabetic characters should not be counted

75

75

Algorithm

```
set  consonantCount  to 0
set  vowelCount  to 0
loop
{
  input  ch
  if  (end of file)
  {
    exit loop
  }

  if  (ch  is a vowel)
  {
    increment  vowelCount
  }
  else if  (ch  is a consonant)
  {
    increment  consonantCount
  }
}
output  consonantCount, vowelCount
```

76

76

Algorithm (cont)

```
set consonantCount to 0
set vowelCount to 0
loop
{
  input ch
  if (end of file)
  {
    exit loop
  }

  if (ch is a vowel)
  {
    increment vowelCount
  }
  else if (ch is a consonant)
  {
    increment consonantCount
  }
}
output consonantCount, vowelCount
```

77

77

Algorithm (cont)

```
set consonantCount to 0
set vowelCount to 0
loop
{
  input ch
  if (end of file)
  {
    exit loop
  }

  if (ch is a vowel)
  {
    increment vowelCount
  }
  else if (ch is a consonant)
  {
    increment consonantCount
  }
}
output consonantCount, vowelCount
```

78

78

Algorithm (cont)

```
set consonantCount to 0
set vowelCount to 0
loop
{
  input ch
  if (end of file)
  {
    exit loop
  }

  if (ch is a vowel)
  {
    increment vowelCount
  }
  else if (ch is a consonant)
  {
    increment consonantCount
  }
}
output consonantCount, vowelCount
```

79

79

Algorithm (cont)

```
set consonantCount to 0
set vowelCount to 0
loop
{
  input ch
  if (end of file)
  {
    exit loop
  }

  if (ch is a vowel)
  {
    increment vowelCount
  }
  else if (ch is a consonant)
  {
    increment consonantCount
  }
}
output consonantCount, vowelCount
```

80

80

Program

```
#include <stdio.h>

int main()
{
   int   consonantCount, vowelCount ;
   char  ch ;

   consonantCount = 0 ;
   vowelCount = 0 ;

   printf("\nInput has  %d  consonants and  %d  vowels.\n",
          consonantCount, vowelCount) ;

   return 0;
}
```

81

81

Program

```
consonantCount = 0 ;
vowelCount = 0 ;

/* For each character in the input, test if it is a
   consonant or a vowel, and adjust the total accordingly */

while ( scanf("%c", &ch) > 0 )
{

}

printf("\nInput has  %d  consonants and  %d  vowels.\n",
       consonantCount, vowelCount) ;
```

82

82

Program

```
/* For each character in the input, test if it is
   a consonant or vowel, and adjust total accordingly */

   while ( scanf("%c", &ch) > 0 )
   {
      if  (ch == 'a' || ch == 'A' ||
           ch == 'e' || ch == 'E' ||
           ch == 'i' || ch == 'I' ||
           ch == 'o' || ch == 'O' ||
           ch == 'u' || ch == 'U')
      {
         /* Vowel */
         vowelCount++ ;
      }
      else if ((ch >= 'a' && ch <= 'z') ||
               (ch >= 'A' && ch <= 'Z'))
      {
         /* Consonant, since vowels already dealt with */
         consonantCount++ ;
      }

   }
```

83

83

```
#include <stdio.h>

/* Count the number of vowels, and the number of consonants in the input */

int main()
{
   int   consonantCount, vowelCount ;
   char  ch ;

   consonantCount = 0 ;
   vowelCount = 0 ;

   /*  For each character in the input, test if it is
       a consonant or vowel, and if so, adjust total accordingly. */

   while ( scanf("%c", &ch) > 0 )
   {
      if  (ch == 'a' || ch == 'A' ||
           ch == 'e' || ch == 'E' ||
           ch == 'i' || ch == 'I' ||
           ch == 'o' || ch == 'O' ||
           ch == 'u' || ch == 'U')
      {
         /* Vowel. */
         vowelCount++ ;
      }
      else if ( (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z') )
      {
         /* Consonant, since vowels already dealt with. */
         consonantCount++ ;
      }
   }

   printf("\nInput has  %d  consonants and  %d  vowels.\n",
          consonantCount, vowelCount) ;

   return 0;
}
```

84

84

Summary

- **`while`** statement
- **`do-while`** statement
- **`for`** statement
- **`break`** statement
- Nested loops
- Infinite loops
- Examples

85

85