

Outline

Modular Design

- multiple choice quiz
- bottom-up design

Python Implementation

- the modules
- main program: quiz.py

The Software Cycle

- quality of product and process
- waterfall and spiral model

Summary + Assignments

MCS 260 Lecture 19
Introduction to Computer Science
Jan Verschelde, 10 October 2007

Modular Design

- multiple choice quiz
- bottom-up design

Python Implementation

- the modules
- main program: quiz.py

The Software Cycle

- quality of product and process
- waterfall and spiral model

Summary + Assignments

Outline

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz

bottom-up design

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: quiz.py

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Summary + Assignments

Multiple Choice Quiz

problem specification

To test our knowledge of computer science concepts, we would like an automatic multiple choice quiz.

Our friend the computer will

1. show a question with multiple possible answers
2. wait for us to make a choice
3. immediately evaluate our choice
4. ask us if we want to continue
5. at the end print how well we did

Wanted: a small but extendable program.

Multiple Choice Quiz

problem specification

To test our knowledge of computer science concepts, we would like an automatic multiple choice quiz.

Our friend the computer will

1. show a question with multiple possible answers
2. wait for us to make a choice
3. immediately evaluate our choice
4. ask us if we want to continue
5. at the end print how well we did

Wanted: a small but extendable program.

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Multiple Choice Quiz

problem specification

To test our knowledge of computer science concepts, we would like an automatic multiple choice quiz.

Our friend the computer will

1. show a question with multiple possible answers
2. wait for us to make a choice
3. immediately evaluate our choice
4. ask us if we want to continue
5. at the end print how well we did

Wanted: a small but extendable program.

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Multiple Choice Quiz

problem specification

To test our knowledge of computer science concepts, we would like an automatic multiple choice quiz.

Our friend the computer will

1. show a question with multiple possible answers
2. wait for us to make a choice
3. immediately evaluate our choice
4. ask us if we want to continue
5. at the end print how well we did

Wanted: a small but extendable program.

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Multiple Choice Quiz

problem specification

To test our knowledge of computer science concepts, we would like an automatic multiple choice quiz.

Our friend the computer will

1. show a question with multiple possible answers
2. wait for us to make a choice
3. immediately evaluate our choice
4. ask us if we want to continue
5. at the end print how well we did

Wanted: a small but extendable program.

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Multiple Choice Quiz

problem specification

To test our knowledge of computer science concepts, we would like an automatic multiple choice quiz.

Our friend the computer will

1. show a question with multiple possible answers
2. wait for us to make a choice
3. immediately evaluate our choice
4. ask us if we want to continue
5. at the end print how well we did

Wanted: a small but extendable program.

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Multiple Choice Quiz

problem specification

To test our knowledge of computer science concepts, we would like an automatic multiple choice quiz.

Our friend the computer will

1. show a question with multiple possible answers
2. wait for us to make a choice
3. immediately evaluate our choice
4. ask us if we want to continue
5. at the end print how well we did

Wanted: a small but extendable program.

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

A Screen Shot

a first prototype

The main program [quiz.py](#), executed at the prompt \$:

```
$ python quiz.py
The octal system has base ...
1. 16
2. 8
3. 10
4. 2
type a number between 1 and 4 : 2
this is the correct answer
continue ? (y/n) n
#correct : 1 #wrong : 0 -> 100.00%
```

Tools for Use

what does Python provide?

Python provides

1. a command line or Python shell environment
2. data structures: lists, tuples, and dictionaries
3. if-elif-else, loops, and functions
4. random number generators in the module `random`
5. our modules will reflect our bottom-up design

What is there at the bottom of our program?

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: `quiz.py`

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Tools for Use

what does Python provide?

Python provides

1. a command line or Python shell environment
2. data structures: lists, tuples, and dictionaries
3. if-elif-else, loops, and functions
4. random number generators in the module `random`
5. our modules will reflect our bottom-up design

What is there at the bottom of our program?

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: `quiz.py`

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Tools for Use

what does Python provide?

Python provides

1. a command line or Python shell environment
2. data structures: lists, tuples, and dictionaries
3. if-elif-else, loops, and functions
4. random number generators in the module `random`
5. our modules will reflect our bottom-up design

What is there at the bottom of our program?

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: `quiz.py`

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Tools for Use

what does Python provide?

Python provides

1. a command line or Python shell environment
2. data structures: lists, tuples, and dictionaries
3. if-elif-else, loops, and functions
4. random number generators in the module `random`
5. our modules will reflect our bottom-up design

What is there at the bottom of our program?

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: `quiz.py`

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Tools for Use

what does Python provide?

Python provides

1. a command line or Python shell environment
2. data structures: lists, tuples, and dictionaries
3. if-elif-else, loops, and functions
4. random number generators in the module `random`
5. our modules will reflect our bottom-up design

What is there at the bottom of our program?

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: `quiz.py`

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Tools for Use

what does Python provide?

Python provides

1. a command line or Python shell environment
2. data structures: lists, tuples, and dictionaries
3. if-elif-else, loops, and functions
4. random number generators in the module `random`
5. our modules will reflect our bottom-up design

What is there at the bottom of our program?

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: `quiz.py`

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Outline

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz

bottom-up design

Modular Design

multiple choice quiz

bottom-up design

Python Implementation

the modules

main program: `quiz.py`

Python Implementation

the modules

main program: `quiz.py`

The Software Cycle

quality of product and process

waterfall and spiral model

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

Summary + Assignments

The Bottom of our Program

MCS 260 L-19

10 October 2007

A multiple choice quiz consists of

1. questions that have short answers;
2. answers, correct ones and variations.

Every question and answer will be a string.

Dictionaries seem very appropriate:

1. the questions are numbered, key = number, the values are strings
2. answers for the corresponding question have the same key as the question
3. answers are dictionaries of dictionaries
4. convention: correct answer comes first

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Bottom of our Program

MCS 260 L-19

10 October 2007

A multiple choice quiz consists of

1. questions that have short answers;
2. answers, correct ones and variations.

Every question and answer will be a string.

Dictionaries seem very appropriate:

1. the questions are numbered, key = number, the values are strings
2. answers for the corresponding question have the same key as the question
3. answers are dictionaries of dictionaries
4. convention: correct answer comes first

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Bottom of our Program

MCS 260 L-19

10 October 2007

A multiple choice quiz consists of

1. questions that have short answers;
2. answers, correct ones and variations.

Every question and answer will be a string.

Dictionaries seem very appropriate:

1. the questions are numbered, key = number, the values are strings
2. answers for the corresponding question have the same key as the question
3. answers are dictionaries of dictionaries
4. convention: correct answer comes first

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Bottom of our Program

MCS 260 L-19

10 October 2007

A multiple choice quiz consists of

1. questions that have short answers;
2. answers, correct ones and variations.

Every question and answer will be a string.

Dictionaries seem very appropriate:

1. the questions are numbered, key = number, the values are strings
2. answers for the corresponding question have the same key as the question
3. answers are dictionaries of dictionaries
4. convention: correct answer comes first

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Bottom of our Program

MCS 260 L-19

10 October 2007

A multiple choice quiz consists of

1. questions that have short answers;
2. answers, correct ones and variations.

Every question and answer will be a string.

Dictionaries seem very appropriate:

1. the questions are numbered, key = number, the values are strings
2. answers for the corresponding question have the same key as the question
3. answers are dictionaries of dictionaries
4. convention: correct answer comes first

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Bottom of our Program

MCS 260 L-19

10 October 2007

A multiple choice quiz consists of

1. questions that have short answers;
2. answers, correct ones and variations.

Every question and answer will be a string.

Dictionaries seem very appropriate:

1. the questions are numbered, key = number, the values are strings
2. answers for the corresponding question have the same key as the question
3. answers are dictionaries of dictionaries
4. convention: correct answer comes first

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Other Modules

MCS 260 L-19

10 October 2007

We separate the functionalities:

1. generating questions and evaluating answers;
2. showing questions and asking answers.

Two modules of functions: questions and dialogue.

Both modules import the Q&A dictionary and export their functions to the main program.

The formulation of questions and answers is centralized in the Q&A dictionary.

A multiple choice question then consists of

1. the number of the question;
2. a permutation of the answers.

In Python we use a tuple, e.g.: (3,[2,1,3,4]).

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Other Modules

MCS 260 L-19

10 October 2007

We separate the functionalities:

1. generating questions and evaluating answers;
2. showing questions and asking answers.

Two modules of functions: questions and dialogue.

Both modules import the Q&A dictionary and export their functions to the main program.

The formulation of questions and answers is centralized in the Q&A dictionary.

A multiple choice question then consists of

1. the number of the question;
2. a permutation of the answers.

In Python we use a tuple, e.g.: (3,[2,1,3,4]).

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Other Modules

MCS 260 L-19

10 October 2007

We separate the functionalities:

1. generating questions and evaluating answers;
2. showing questions and asking answers.

Two modules of functions: questions and dialogue.

Both modules import the Q&A dictionary and export their functions to the main program.

The formulation of questions and answers is centralized in the Q&A dictionary.

A multiple choice question then consists of

1. the number of the question;
2. a permutation of the answers.

In Python we use a tuple, e.g.: (3,[2,1,3,4]).

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Other Modules

MCS 260 L-19

10 October 2007

We separate the functionalities:

1. generating questions and evaluating answers;
2. showing questions and asking answers.

Two modules of functions: questions and dialogue.

Both modules import the Q&A dictionary and export their functions to the main program.

The formulation of questions and answers is centralized in the Q&A dictionary.

A multiple choice question then consists of

1. the number of the question;
2. a permutation of the answers.

In Python we use a tuple, e.g.: (3,[2,1,3,4]).

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

The Other Modules

We separate the functionalities:

1. generating questions and evaluating answers;
2. showing questions and asking answers.

Two modules of functions: questions and dialogue.

Both modules import the Q&A dictionary and export their functions to the main program.

The formulation of questions and answers is centralized in the Q&A dictionary.

A multiple choice question then consists of

1. the number of the question;
2. a permutation of the answers.

In Python we use a tuple, e.g.: (3,[2,1,3,4]).

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

The Other Modules

We separate the functionalities:

1. generating questions and evaluating answers;
2. showing questions and asking answers.

Two modules of functions: questions and dialogue.

Both modules import the Q&A dictionary and export their functions to the main program.

The formulation of questions and answers is centralized in the Q&A dictionary.

A multiple choice question then consists of

1. the number of the question;
2. a permutation of the answers.

In Python we use a tuple, e.g.: (3,[2,1,3,4]).

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

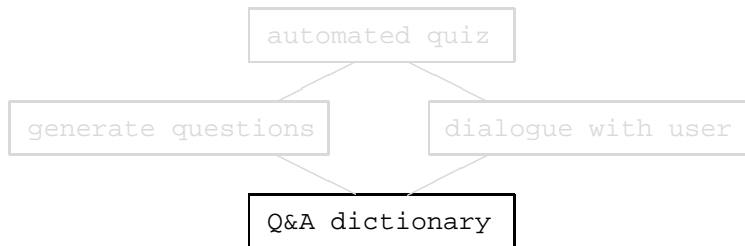
Summary + Assignments

The Modular Design

MCS 260 L-19

10 October 2007

At the bottom: the Q&A dictionary.
Two libraries of functions: questions and dialogue.



Information Hiding:

the main program does not import the Q&A dictionary.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

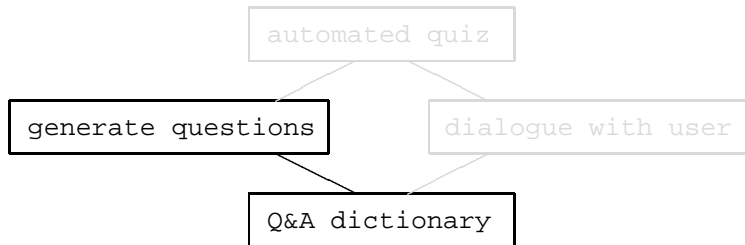
The Modular Design

MCS 260 L-19

10 October 2007

At the bottom: the Q&A dictionary.

Two libraries of functions: questions and dialogue.



Information Hiding:

the main program does not import the Q&A dictionary.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

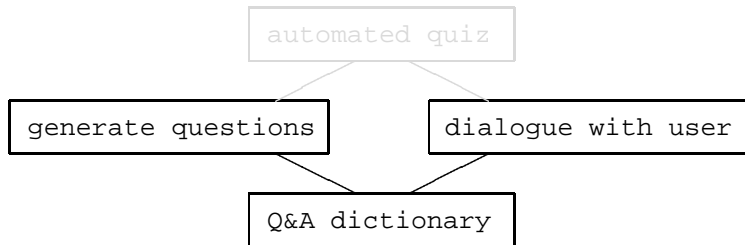
The Modular Design

MCS 260 L-19

10 October 2007

At the bottom: the Q&A dictionary.

Two libraries of functions: questions and dialogue.



Information Hiding:

the main program does not import the Q&A dictionary.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

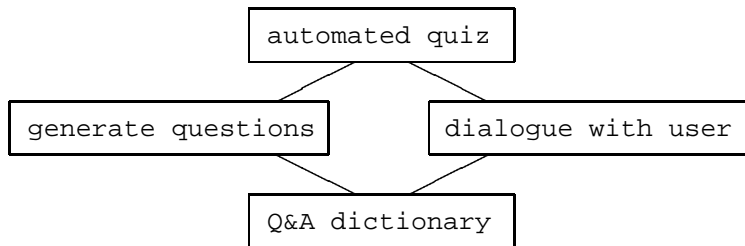
Summary +
Assignments

The Modular Design

MCS 260 L-19

10 October 2007

At the bottom: the Q&A dictionary.
Two libraries of functions: questions and dialogue.



Information Hiding:

the main program does not import the Q&A dictionary.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

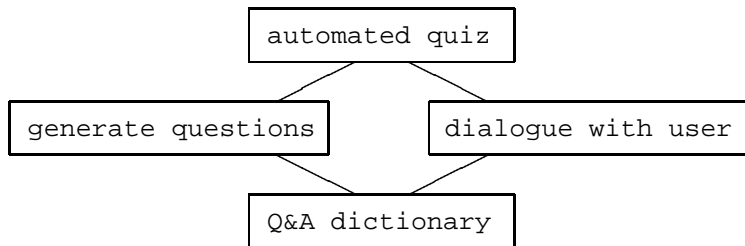
The Modular Design

MCS 260 L-19

10 October 2007

At the bottom: the Q&A dictionary.

Two libraries of functions: questions and dialogue.



Information Hiding:

the main program does not import the Q&A dictionary.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Outline

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules

main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

The Q&A Dictionary

in the file `quand.py`

```
questions = { \
    1:'How many bits go in one byte ?', \
    2:'Who is the main developer of Python ?', \
    3:'Which statement is true ?', \
    4:'The octal system has base ...' \
}

answers = { \
    1:{ 1:'8', 2:'4', 3:'2', 4:'16' }, \
    2:{ 1:'Guido van Rossum', 2:'John von Neumann', \
        3:'Rashi Gupta', 4:'John Backus' }, \
    3:{ 1:'a compiler translates source code', \
        2:'a compiler executes object code', \
        3:'a compiler formats source code', \
        4:'a compiler stores code on file' }, \
    4:{ 1:'8', 2:'2', 3:'10', 4:'16' } \
}
```

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: `quiz.py`

The Software Cycle

quality of product and
process
waterfall and spiral model

Summary + Assignments

The Q&A Dictionary

in the file `quand.py`

```
questions = { \
    1:'How many bits go in one byte ?', \
    2:'Who is the main developer of Python ?', \
    3:'Which statement is true ?', \
    4:'The octal system has base ...' \
}

answers = { \
    1:{ 1:'8', 2:'4', 3:'2', 4:'16' }, \
    2:{ 1:'Guido van Rossum', 2:'John von Neumann', \
        3:'Rashi Gupta', 4:'John Backus' }, \
    3:{ 1:'a compiler translates source code', \
        2:'a compiler executes object code', \
        3:'a compiler formats source code', \
        4:'a compiler stores code on file' }, \
    4:{ 1:'8', 2:'2', 3:'10', 4:'16' } \
}
```

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: `quiz.py`

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Specification of questions.py

MCS 260 L-19

10 October 2007

`help(questions)` shows under FUNCTIONS:

`evaluate(L, c)`

evaluates choice `c` to True or False
where `L` is a list of answers

`generate()`

returns a multiple choice question
as a tuple `(q,L)`, where `q` is a number,
and `L` is a list of numbers with answers

`shuffle(L)`

shuffles the order of the answers
on return is a permutation of the list `L`

`update_tally(t, b)`

updates tally of True or False answers
where `t` is a list of two elements and
`b` is True or False, returns the updated tally

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Specification of questions.py

MCS 260 L-19

10 October 2007

`help(questions)` shows under FUNCTIONS:

`evaluate(L, c)`

evaluates choice `c` to True or False
where `L` is a list of answers

`generate()`

returns a multiple choice question
as a tuple `(q,L)`, where `q` is a number,
and `L` is a list of numbers with answers

`shuffle(L)`

shuffles the order of the answers
on return is a permutation of the list `L`

`update_tally(t, b)`

updates tally of True or False answers
where `t` is a list of two elements and
`b` is True or False, returns the updated tally

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Specification of questions.py

`help(questions)` shows under FUNCTIONS:

`evaluate(L, c)`

evaluates choice `c` to True or False
where `L` is a list of answers

`generate()`

returns a multiple choice question
as a tuple `(q,L)`, where `q` is a number,
and `L` is a list of numbers with answers

`shuffle(L)`

shuffles the order of the answers
on return is a permutation of the list `L`

`update_tally(t, b)`

updates tally of True or False answers
where `t` is a list of two elements and
`b` is True or False, returns the updated tally

Specification of questions.py

`help(questions)` shows under FUNCTIONS:

`evaluate(L, c)`

evaluates choice `c` to True or False
where `L` is a list of answers

`generate()`

returns a multiple choice question
as a tuple `(q,L)`, where `q` is a number,
and `L` is a list of numbers with answers

`shuffle(L)`

shuffles the order of the answers
on return is a permutation of the list `L`

`update_tally(t, b)`

updates tally of True or False answers
where `t` is a list of two elements and
`b` is True or False, returns the updated tally

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Specification of dialogue.py

MCS 260 L-19

10 October 2007

`help(dialogue)` shows under FUNCTIONS:

```
prompt_choice(n)
```

```
    asks the user for a choice  
    where n is the number of choices
```

```
show_outcome(b)
```

```
    reports outcome to user  
    where b is True or False
```

```
show_question(q, L)
```

```
    displays the questions and the answers  
    on input is the number q of the question  
    and the list L of answers
```

```
show_tally(t)
```

```
    shows final tally, t is list
```

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Specification of dialogue.py

MCS 260 L-19

10 October 2007

`help(dialogue)` shows under FUNCTIONS:

```
prompt_choice(n)
```

```
    asks the user for a choice  
    where n is the number of choices
```

```
show_outcome(b)
```

```
    reports outcome to user  
    where b is True or False
```

```
show_question(q, L)
```

```
    displays the questions and the answers  
    on input is the number q of the question  
    and the list L of answers
```

```
show_tally(t)
```

```
    shows final tally, t is list
```

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Specification of dialogue.py

MCS 260 L-19

10 October 2007

`help(dialogue)` shows under FUNCTIONS:

```
prompt_choice(n)
```

asks the user for a choice
where n is the number of choices

```
show_outcome(b)
```

reports outcome to user
where b is True or False

```
show_question(q, L)
```

displays the questions and the answers
on input is the number q of the question
and the list L of answers

```
show_tally(t)
```

shows final tally, t is list

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Specification of dialogue.py

`help(dialogue)` shows under FUNCTIONS:

```
prompt_choice(n)
```

asks the user for a choice
where n is the number of choices

```
show_outcome(b)
```

reports outcome to user
where b is True or False

```
show_question(q, L)
```

displays the questions and the answers
on input is the number q of the question
and the list L of answers

```
show_tally(t)
```

shows final tally, t is list

Outline

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: `quiz.py`

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: `quiz.py`

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

The Main Program

in the file quiz.py

```
import questions
import dialogue

t = [0,0]
while True:
    (q,a) = questions.generate()
    s = questions.shuffle(a)
    dialogue.show_question(q,s)
    c = dialogue.prompt_choice(len(s))
    e = questions.evaluate(s,c)
    dialogue.show_outcome(e)
    t = questions.update_tally(t,e)
    more = raw_input('continue ? (y/n) ')
    if more != 'y': break
dialogue.show_tally(t)
```

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and
process
waterfall and spiral model

Summary + Assignments

The Main Program

in the file quiz.py

```
import questions
import dialogue

t = [0,0]
while True:
    (q,a) = questions.generate()
    s = questions.shuffle(a)
    dialogue.show_question(q,s)
    c = dialogue.prompt_choice(len(s))
    e = questions.evaluate(s,c)
    dialogue.show_outcome(e)
    t = questions.update_tally(t,e)
    more = raw_input('continue ? (y/n) ')
    if more != 'y': break
dialogue.show_tally(t)
```

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and
process
waterfall and spiral model

Summary + Assignments

The Main Program

in the file quiz.py

```
import questions
import dialogue

t = [0,0]
while True:
    (q,a) = questions.generate()
    s = questions.shuffle(a)
    dialogue.show_question(q,s)
    c = dialogue.prompt_choice(len(s))
    e = questions.evaluate(s,c)
    dialogue.show_outcome(e)
    t = questions.update_tally(t,e)
    more = raw_input('continue ? (y/n) ')
    if more != 'y': break
dialogue.show_tally(t)
```

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and
process
waterfall and spiral model

Summary + Assignments

The Main Program

in the file quiz.py

```
import questions
import dialogue

t = [0,0]
while True:
    (q,a) = questions.generate()
    s = questions.shuffle(a)
    dialogue.show_question(q,s)
    c = dialogue.prompt_choice(len(s))
    e = questions.evaluate(s,c)
    dialogue.show_outcome(e)
    t = questions.update_tally(t,e)
    more = raw_input('continue ? (y/n) ')
    if more != 'y': break
dialogue.show_tally(t)
```

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

The Main Program

in the file quiz.py

```
import questions
import dialogue

t = [0,0]
while True:
    (q,a) = questions.generate()
    s = questions.shuffle(a)
    dialogue.show_question(q,s)
    c = dialogue.prompt_choice(len(s))
    e = questions.evaluate(s,c)
    dialogue.show_outcome(e)
    t = questions.update_tally(t,e)
    more = raw_input('continue ? (y/n) ')
    if more != 'y': break
dialogue.show_tally(t)
```

MCS 260 L-19

10 October 2007

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and
process
waterfall and spiral model

Summary + Assignments

Two Functions in questions.py

```
import quand

def generate():
    "returns a multiple choice question\
\nas a tuple (q,L), where q is a number,\
\nand L is a list of numbers with answers"
    import random
    q = random.randint(1,len(quand.questions))
    L = range(1,len(quand.answers[q])+1)
    return (q,L)

def evaluate(L,c):
    "evaluates choice c to True or False\
\nwhere L is a list of answers"
    return (L[c-1] == 1)
```

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Two Functions in questions.py

```
import quand

def generate():
    "returns a multiple choice question\
\nas a tuple (q,L), where q is a number,\
\nand L is a list of numbers with answers"
    import random
    q = random.randint(1,len(quand.questions))
    L = range(1,len(quand.answers[q])+1)
    return (q,L)

def evaluate(L,c):
    "evaluates choice c to True or False\
\nwhere L is a list of answers"
    return (L[c-1] == 1)
```

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and
process
waterfall and spiral model

Summary + Assignments

Two Functions in dialogue.py

```
import quand

def show_question(q,L):
    "displays the questions and the answers\
\non input is the number q of the question\
\nand the list L of answers"
    print quand.questions[q]
    choice = 1
    for answer in L:
        p = '%d' % choice + '. '
        p = p + quand.answers[q][answer]
        print p
        choice = choice + 1

def show_outcome(b):
    "reports outcome to user\
\nwhere b is True or False"
    if b: print 'this is the correct answer'
    else: print 'wrong, please try again'
```

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Two Functions in dialogue.py

```
import quand

def show_question(q,L):
    "displays the questions and the answers\
\non input is the number q of the question\
\nand the list L of answers"
    print quand.questions[q]
    choice = 1
    for answer in L:
        p = '%d' % choice + '. '
        p = p + quand.answers[q][answer]
        print p
        choice = choice + 1

def show_outcome(b):
    "reports outcome to user\
\nwhere b is True or False"
    if b: print 'this is the correct answer'
    else: print 'wrong, please try again'
```

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

We are experiencing **the software crisis**:

→ *the industrial sector can not keep up with the ever growing demands of the market.*

Symptoms:

1. products are delivered late
2. the cost exceed budgets
3. too many bugs appear

The relative cost of hardware versus software

- ▶ was 80% versus 20% in the 1960s
- ▶ became 20% versus 80% in the 1980s
- ▶ keeps going in the same direction.

Reason: hardware production is technology intensive, whereas software *creation* is human intensive.

We are experiencing **the software crisis**:

→ *the industrial sector can not keep up with the ever growing demands of the market.*

Symptoms:

1. products are delivered late
2. the cost exceed budgets
3. too many bugs appear

The relative cost of hardware versus software

- ▶ was 80% versus 20% in the 1960s
- ▶ became 20% versus 80% in the 1980s
- ▶ keeps going in the same direction.

Reason: hardware production is technology intensive, whereas software *creation* is human intensive.

The Software Cycle

MCS 260 L-19

10 October 2007

We are experiencing **the software crisis**:

→ *the industrial sector can not keep up with the ever growing demands of the market.*

Symptoms:

1. products are delivered late
2. the cost exceed budgets
3. too many bugs appear

The relative cost of hardware versus software

- ▶ was 80% versus 20% in the 1960s
- ▶ became 20% versus 80% in the 1980s
- ▶ keeps going in the same direction.

Reason: hardware production is technology intensive, whereas software *creation* is human intensive.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

We are experiencing **the software crisis**:

→ *the industrial sector can not keep up with the ever growing demands of the market.*

Symptoms:

1. products are delivered late
2. the cost exceed budgets
3. too many bugs appear

The relative cost of hardware versus software

- ▶ was 80% versus 20% in the 1960s
- ▶ became 20% versus 80% in the 1980s
- ▶ keeps going in the same direction.

Reason: hardware production is technology intensive, whereas software *creation* is human intensive.

We are experiencing **the software crisis**:

→ *the industrial sector can not keep up with the ever growing demands of the market.*

Symptoms:

1. products are delivered late
2. the cost exceed budgets
3. too many bugs appear

The relative cost of hardware versus software

- ▶ was 80% versus 20% in the 1960s
- ▶ became 20% versus 80% in the 1980s
- ▶ keeps going in the same direction.

Reason: hardware production is technology intensive, whereas software *creation* is human intensive.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

We are experiencing **the software crisis**:

→ *the industrial sector can not keep up with the ever growing demands of the market.*

Symptoms:

1. products are delivered late
2. the cost exceed budgets
3. too many bugs appear

The relative cost of hardware versus software

- ▶ was 80% versus 20% in the 1960s
- ▶ became 20% versus 80% in the 1980s
- ▶ keeps going in the same direction.

Reason: hardware production is technology intensive, whereas software *creation* is human intensive.

Outline

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Quality of Software Design

industrial quality

Industrial quality refers to products

1. that must meet a target value on average,
2. and within a tolerable standard deviation.

example: a search engine should give one page of relevant answers within an average of 10 seconds, and standard deviation of 2 seconds.

We distinguish between quality of product and process:

1. a high quality product meets user's expectations,
2. also the quality of the production process matters!

External qualities can be observed directly by the user.

example: easy to use

Internal qualities can be measured only by the producer.

example: design documentation

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Quality of Software Design

industrial quality

Industrial quality refers to products

1. that must meet a target value on average,
2. and within a tolerable standard deviation.

example: a search engine should give one page of relevant answers within an average of 10 seconds, and standard deviation of 2 seconds.

We distinguish between quality of product and process:

1. a high quality product meets user's expectations,
2. also the quality of the production process matters!

External qualities can be observed directly by the user.

example: easy to use

Internal qualities can be measured only by the producer.

example: design documentation

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Quality of Software Design

industrial quality

Industrial quality refers to products

1. that must meet a target value on average,
2. and within a tolerable standard deviation.

example: a search engine should give one page of relevant answers within an average of 10 seconds, and standard deviation of 2 seconds.

We distinguish between quality of product and process:

1. a high quality product meets user's expectations,
2. also the quality of the production process matters!

External qualities can be observed directly by the user.

example: easy to use

Internal qualities can be measured only by the producer.

example: design documentation

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Quality of Software Design

industrial quality

Industrial quality refers to products

1. that must meet a target value on average,
2. and within a tolerable standard deviation.

example: a search engine should give one page of relevant answers within an average of 10 seconds, and standard deviation of 2 seconds.

We distinguish between quality of product and process:

1. a high quality product meets user's expectations,
2. also the quality of the production process matters!

External qualities can be observed directly by the user.

example: easy to use

Internal qualities can be measured only by the producer.

example: design documentation

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Quality of Software Design

industrial quality

Industrial quality refers to products

1. that must meet a target value on average,
2. and within a tolerable standard deviation.

example: a search engine should give one page of relevant answers within an average of 10 seconds, and standard deviation of 2 seconds.

We distinguish between quality of product and process:

1. a high quality product meets user's expectations,
2. also the quality of the production process matters!

External qualities can be observed directly by the user.

example: easy to use

Internal qualities can be measured only by the producer.

example: design documentation

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Quality of Software Design

industrial quality

Industrial quality refers to products

1. that must meet a target value on average,
2. and within a tolerable standard deviation.

example: a search engine should give one page of relevant answers within an average of 10 seconds, and standard deviation of 2 seconds.

We distinguish between quality of product and process:

1. a high quality product meets user's expectations,
2. also the quality of the production process matters!

External qualities can be observed directly by the user.

example: easy to use

Internal qualities can be measured only by the producer.

example: design documentation

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Important Software Quality Features

what is good software?

A short list of important criteria:

correctness must satisfy requirements

How reliable is the software?

efficiency economical use of resources such as hardware and time

Do we understand how the complexity of the problem and the software relate?

modifiability to different environments and requirements

How hard is it to correct errors or to add new features?

reusability of the same software in different applications

Mathematical libraries are a typical example of reusable software.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Important Software Quality Features

what is good software?

A short list of important criteria:

correctness must satisfy requirements

How reliable is the software?

efficiency economical use of resources such as hardware and time

Do we understand how the complexity of the problem and the software relate?

modifiability to different environments and requirements

How hard is it to correct errors or to add new features?

reusability of the same software in different applications

Mathematical libraries are a typical example of reusable software.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Important Software Quality Features

what is good software?

A short list of important criteria:

correctness must satisfy requirements

How reliable is the software?

efficiency economical use of resources such as hardware and time

Do we understand how the complexity of the problem and the software relate?

modifiability to different environments and requirements

How hard is it to correct errors or to add new features?

reusability of the same software in different applications

Mathematical libraries are a typical example of reusable software.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Important Software Quality Features

what is good software?

A short list of important criteria:

correctness must satisfy requirements

How reliable is the software?

efficiency economical use of resources such as hardware and time

Do we understand how the complexity of the problem and the software relate?

modifiability to different environments and requirements

How hard is it to correct errors or to add new features?

reusability of the same software in different applications

Mathematical libraries are a typical example of reusable software.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments

Outline

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process
waterfall and spiral model

Summary + Assignments

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process

waterfall and spiral model

Summary +
Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Waterfall Model

seven phases in sequence

The software life cycle model describes the software production in seven phases:

1. feasibility study: cost-benefit analysis
2. system requirements specification: what must it do?
3. system architecture design: modules and relations
4. implementation and verification of modules
5. system integration and verification
6. installation: deliver software to the user
7. maintenance: fixing bugs and adding new features

As the phases occur in a strict sequence, just as water never flows upwards, this model is like a waterfall.

This deductive or top-down description of the life cycle has its merits, especially with feedback.

Modular Design

multiple choice quiz
bottom-up design

Python Implementation

the modules
main program: quiz.py

The Software Cycle

quality of product and process

waterfall and spiral model

Summary + Assignments

The Spiral Model

MCS 260 L-19

10 October 2007

Four sectors in the software development:

1. requirements analysis and specification
2. system architecture design
3. implementation
4. verification

Incremental development of software:

1. think of a product with limited scope
2. create the design of the first product
3. implement a prototype
4. test and deliver the prototype

Based on feedback, enlarge the specifications, revise the designs, build new prototypes, test and deliver again.

As the cycles over the four sectors get larger, the complexity of the software grows.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process

waterfall and spiral model

Summary +
Assignments

The Spiral Model

MCS 260 L-19

10 October 2007

Four sectors in the software development:

1. requirements analysis and specification
2. system architecture design
3. implementation
4. verification

Incremental development of software:

1. think of a product with limited scope
2. create the design of the first product
3. implement a prototype
4. test and deliver the prototype

Based on feedback, enlarge the specifications, revise the designs, build new prototypes, test and deliver again.

As the cycles over the four sectors get larger, the complexity of the software grows.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process

waterfall and spiral model

Summary +
Assignments

The Spiral Model

MCS 260 L-19

10 October 2007

Four sectors in the software development:

1. requirements analysis and specification
2. system architecture design
3. implementation
4. verification

Incremental development of software:

1. think of a product with limited scope
2. create the design of the first product
3. implement a prototype
4. test and deliver the prototype

Based on feedback, enlarge the specifications, revise the designs, build new prototypes, test and deliver again.

As the cycles over the four sectors get larger, the complexity of the software grows.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process

waterfall and spiral model

Summary +
Assignments

The Spiral Model

MCS 260 L-19

10 October 2007

Four sectors in the software development:

1. requirements analysis and specification
2. system architecture design
3. implementation
4. verification

Incremental development of software:

1. think of a product with limited scope
2. create the design of the first product
3. implement a prototype
4. test and deliver the prototype

Based on feedback, enlarge the specifications, revise the designs, build new prototypes, test and deliver again.

As the cycles over the four sectors get larger, the complexity of the software grows.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process

waterfall and spiral model

Summary +
Assignments

Summary + Assignments

MCS 260 L-19

10 October 2007

We covered in the lecture:

- ▶ more of chapter 6 in *Making Use of Python*;
- ▶ chapter 20 in *The Art & Craft of Computing*.

Assignments:

1. Enlarge the Q&A answer dictionary with a least six more questions about relevant CS concepts.
2. Modify the program so the user has the option to see the correct answer when wrong.
3. Design your own trivia quiz.

Homework will be collected on Monday 15 October:

exercises 1 and 3 of Lecture 14; exercise 1 of Lecture 15; and exercises 1 and 2 from above.

Modular Design

multiple choice quiz
bottom-up design

Python
Implementation

the modules
main program: quiz.py

The Software
Cycle

quality of product and
process
waterfall and spiral model

Summary +
Assignments