Review Questions

Page 170-171

Multiple Choice

1. A _____ structure can execute a set of statements only under certain circumstances.
   a. sequence
   b. circumstantial
   ✓c. decision
   d. Boolean

2. A _____ structure provides one alternative path of execution.
   a. sequence
   ✓b. single alternative decision
   c. one path alternative
   d. single execution decision

7. You use a(n) _____ statement to write a dual alternative decision structure.
   a. `test-jump`
   b. `if`
   ✓c. `if-else`
   d. `if-call`

11. The _____ operator takes a Boolean expression as its operand and reverses its logical value.
   a. `and`
   b. `or`
   ✓c. `not`
   d. `either`

Algorithm Workbench

Page 172-173

2. Write an `if` statement that assigns 10 to the variable `b`, and 50 to the variable `c` if the variable `a` is equal to 100.

```
if a == 100:
        b=10
        c=50
```

6. Write an `if-else` statement that assigns `True` to the `again` variable if the `score` variable is within the range of 40 to 49. If the `score` variable's value is outside this range, assign `False` to the `again` variable.

```
if score >= 40 and score <= 49:
        again = True
else:
        again=False
```

7. Write an `if-else` statement that determines whether the `points` variable is outside the range of 9 to 51. If the variable's value is outside this range it should display "Invalid points." Otherwise, it should display "Valid points."

```
if points < 9 or points > 51:
        print ( 'Invalid points.' )
else:
        print ( 'Valid points.' )
```

8. Write an `if` statement that uses the turtle graphics library to determine whether the turtle's heading is in the range of 0 degrees to 45 degrees (including 0 and 45 in the range). If so, raise the turtle's pen.

```
if turtle.heading >= 0 and turtle.heading <= 45:
        turtle.penup()
```

Review Questions

Multiple Choice

Page 223

5. A(n) _____ loop has no way of ending and repeats until the program is interrupted.
   a. indeterminate
   b. interminable
   c. infinite
   d. timeless
6. The -= operator is an example of a(n) _____ operator.
   a. relational
   b. augmented assignment
   c. complex assignment
   d. reverse assignment
7. A(n) _____ variable keeps a running total.
   a. sentinel
   b. sum
   c. total
   d. accumulator
8. A(n) _____ is a special value that signals when there are no more items from a list of items to be processed. This value cannot be mistaken as an item from the list.
   a. sentinel
   b. flag
   c. signal
   d. accumulator

Algorithm Workbench

Page 225

5. Write a loop that calculates the total of the following series of numbers:

$$\frac{1}{30} + \frac{2}{29} + \frac{3}{28} + \ldots \frac{30}{1}$$

```
sum=0.0
for i in range(1,30):
        sum+=(i/(31-i))
```

6. Rewrite the following statements using augmented assignment operators.

a. `x = x + 1` `x=+1`
b. `x = x * 2` `x*=2`
c. `x = x / 10` `x/=10`
d. `x = x - 100` `x-=100`

7. Write a set of nested loops that displays the first ten values of the multiplication tables from 1 to 10. The code should print 100 lines in total, starting at "1 × 1 = 1" and ending with "10 × 10 = 100".

```
for i in range(1,10):
      for j in range(1,10):
            print( i,'X',j,'=',i*j)
```

9. Write code that prompts the user to enter a number in the range of 1 through 100 and validates the input.

```
num=(inputint('Enter a number 1-100:'))
while ( num < 1 or num > 100):
       num=int(input('Number must be in the range 1-100:'))
```

Checkpoint Page 247

5.10 What is a local variable? How is access to a local variable restricted?

Any variable created inside a function is a local variable. A local variable is created inside a function and cannot be accessed by any statement that are outside the function.

5.11 What is a variable's scope?

Scope of a variable is the part of a program in which a variable may be accessed. A local variable's scope is the function in which the variable is created.

5.12 Is it permissible for a local variable in one function to have the same name as a local variable in a different function?

Yes, it is. Different functions can have local variables with the same names because the functions cannot see each other's local variables.

Checkpoint Page 257

Checkpoint

5.13 What are the pieces of data that are passed into a function called?
Arguments

5.14 What are the variables that receive pieces of data in a function called?
Parameter variables

5.15 What is a parameter variable's scope?
Parameter variables have a function scope. They only can be accessed within the function. They are local variables in the function they are created.

5.16 When a parameter is changed, does this affect the argument that was passed into the parameter?
No, it doesn't. Because parameter variables are local variables and have a function scope.

5.17 The following statements call a function named show_data. Which of the statements passes arguments by position, and which passes keyword arguments?
a. show_data(name='Kathryn', age=25) **->Passing arguments by keywords**
b. show_data('Kathryn', 25) **-> Passing arguments by position**

Checkpoint — Page 271

5.23 Why are library functions like "black boxes"?
Because we use them without looking into it in detail.
5.24 What does the following statement do?
```
x = random.randint(1, 100)
```
Generates an integer number from 1 to 99 inclusively and assign it to x variable.
5.25 What does the following statement do?
```
print(random.randint(1, 20))
```
Generates an integer number from 1 to 19 inclusively and displays the result.
5.26 What does the following statement do?
```
print(random.randrange(10, 20))
```
Generates an integer number from 10 to 19 inclusively and displays the result.
5.27 What does the following statement do?
```
print(random.random())
```
Generates a float number 0 to 1 (excluding 1.0)
5.28 What does the following statement do?
```
print(random.uniform(0.1, 0.5))
```
Generates a float number 0.1 to 0.5
5.29 When the random module is imported, what does it use as a seed value for random number generation?
By default it uses the time of the computer
5.30 What happens if the same seed value is always used for generating random numbers?
Same seed value gives the same results at any run

Checkpoint — Page 283

5.31 What is the purpose of the `return` statement in a function?
To return data to the caller. It can be used for many purposes.

5.32 Look at the following function definition:
```
def do_something(number):
      return number * 2
```
a. What is the name of the function?
`do_something`
b. What does the function do?
It returns the double of the number sent by the caller back to the caller.
c. Given the function definition, what will the following statement display?
```
      print(do_something(10))
```
20

5.33 What is a Boolean function?
A Function returns True or False – Boolean variable to the caller.

Review Questions

Page 297-298

Multiple Choice

3. The first line of a function definition is known as the _____.
   a. body
   b. introduction
   c. initialization
   d. header
4. You _____ a function to execute it.
   a. define
   b. call
   c. import
   d. export
6. A _____ is a diagram that gives a visual representation of the relationships between functions in a program.
   a. flowchart
   b. function relationship chart
   c. symbol chart
   d. hierarchy chart
10. A(n) _____ is a special variable that receives a piece of data when a function is called.
    a. argument
    b. parameter
    c. header
    d. packet
11. A variable that is visible to every function in a program file is a _____.
    a. local variable
    b. universal variable
    c. program-wide variable
    d. global variable

Review Questions

Page 299

Multiple Choice

12. When possible, you should avoid using _____ variables in a program.
    a. local
    b. global
    c. reference
    d. parameter
14. This standard library function returns a random integer within a specified range of values.
    a. `random`
    b. `randint`
    c. `random_integer`
    d. `uniform`
15. This standard library function returns a random floating-point number in the range of 0.0 up to 1.0 (but not including 1.0).
    a. `random`
    b. `randint`
    c. `random_integer`
    d. `uniform`
17. This statement causes a function to end and sends a value back to the part of the program that called the function.
    a. `end`
    b. `send`
    c. `exit`
    d. `return`
18. This is a design tool that describes the input, processing, and output of a function.
    a. hierarchy chart
    b. IPO chart
    c. datagram chart
    d. data processing chart

Algorithm Workbench

Page 302

9. Write a function named `times_ten` that accepts a number as an argument. When the function is called, it should return the value of its argument multiplied times 10.

```
def times_ten(number):
    return number * 10
```

10. Write a function named `is_valid_length` that accepts a string and an integer as arguments. When the function is called, it should return False if the length of the string is greater than the integer, otherwise it should return True.

```
def is_valid_length(text, limit):
    if len(text) > limit:
        return False
    else:
        return True
```

Checkpoint

Page 326

6.1 – 6.2 What is an output file and input file?
an output file that data is written to and an intput file from which data is read
6.3 What three steps must be taken by a program when it uses a file?
Opening the file, Processing the file and Closing the file.
6.4 In general, what are the two types of files? What is the difference between these two types of files?
Text and Binary files. In the text files, content is encoded as text so we browse by using text editors like Notepad.
6.5 What are the two types of file access? What is the difference between these two?
Sequential and Direct Access. Sequential access is done processing line by line. In the DA, you can jump to any data.
6.6 When writing a program that performs an operation on a file, what two file associated names do you have to work with in your code?
filename and file_object. We use file_object associated with the file to access to the file.
6.7 If a file already exists, what happens to it if you try to open it as an output file (using the 'w' mode)?
If the file already, then 'w' mode erases the content of the file.
6.8 What is the purpose of opening a file?
To access a file for processing, we need to open the file.
6.9 What is the purpose of closing a file?
At the end, we must close the file. Closing a file disconnects the file from the program.
6.10 What is a file's read position? Initially, where is the read position when an input file is opened?
A file's read position marks the location of the next item that will be read from the file. Initially, the read position is set to the beginning of the file.
6.11 In what mode do you open a file if you want to write data to it, but you do not want to erase the file's existing contents? When you write data to such a file, to what part of the file is the data written?
In 'a' mode, we open a file to be written to. All data written to the file is appended to its end.

Checkpoint — Page 333

6.12 Write a short program that uses a for loop to write the numbers 1 through 10 to a file.

```
for i in range(1,11):
        file_object.write(str(i)+'\n')
```

6.13 What does it mean when the `readline` method returns an empty string?

it means the end of the file is reached.

6.14 Assume the file `data.txt` exists and contains several lines of text. Write a short program using the while loop that displays each line in the file.

```
data_file = open(data.txt,'r')
text=data_file.readline()
while text != '':
        text=text.rstrip('\n')
        print(text)
        text=data_file.readline()   # Reading the next line
```

6.15 Revise the program that you wrote for **Checkpoint 6.14** to use the for loop instead of the while loop.

```
data_file = open(data.txt,'r')
for line in data_file:
        line=line.rstrip('\n')
        print(line)
```

Checkpoint — Page 359

6.20 If an exception is raised and the program does not handle it with a try/except statement, what happens?

Program halts abruptly with traceback error message.

6.21 What type of exception does a program raise when it tries to open a nonexistent file?

If we try to open a non-existent file in 'r' mode then the program raises `FileNotFoundError` exception.

6.22 What type of exception does a program raise when it uses the float function to convert a non-numeric string to a number?

it raises a `ValueError` exception.

Review Questions

Page 359-360-361

Multiple Choice

3. Before a file can be used by a program, it must be _____.
   a. formatted
   b. encrypted
   c. closed
   d. opened

7. When working with this type of file, you access its data from the beginning of the file to the end of the file.
   a. ordered access
   b. binary access
   c. direct access
   d. sequential access

12. This is a single piece of data within a record.
   d. field
   b. variable
   c. delimiter
   d. subrecord

13. When an exception is generated, it is said to have been _____.
   a. built
   b. raised
   c. caught
   d. killed

15. You write this statement to respond to exceptions.
   a. `run/handle`
   b. `try/except`
   c. `try/handle`
   d. `attempt/except`

Algorithm Workbench

Page 362

6. Write code that opens an output file with the filename `number_list.txt`, but does not erase the file's contents if it already exists.

```
outfile = open( number_list.txt, 'r' )
```

or

```
outfile = open( number_list.txt, 'a' )
```

7. A file exists on the disk named `students.txt`. The file contains several records, and each record contains two fields: (1) the student's name, and (2) the student's score for the final exam. Write code that deletes the record containing "John Perz" as the student name.

```
std_file = open('students.txt', 'r')
 temp_file = open('temp.txt', 'w')
name = std_file.readline()
while name != '':
        score = std_file.readline()
                name = name.rstrip('\n')    #Strip '\n'
        if name != 'John Perz' :  # Don't Write this data to temp.txt
                temp_file.write(name+'\n')
                temp_file.write(score)
        name = std_file.readline()    #read the name in the next record
std_file.close()
temp_file.close()
os.remove('employees.txt')
os.rename('temp.txt', 'employees.txt')
```

Review Questions

Page 421-422

Multiple Choice

1. This term refers to an individual item in a list.
   a. element
   b. bin
   c. cubbyhole
   d. slot
4. This is the last index in a list.
   a. 1
   b. 99
   c. 0
   d. The size of the list minus one
5. This will happen if you try to use an index that is out of range for a list.
   a. A `ValueError` exception will occur.
   b. An `IndexError` exception will occur.
   c. The list will be erased and the program will continue to run.
   d. Nothing—the invalid index will be ignored.
7. When the * operator's left operand is a list and its right operand is an integer, the operator becomes this.
   a. The multiplication operator
   b. The repetition operator
   c. The initialization operator
   d. Nothing—the operator does not support those types of operands.
11. If you call the `index` method to locate an item in a list and the item is not found, this happens.
   a. A `ValueError` exception is raised.
   b. An `InvalidIndex` exception is raised.
   c. The method returns −1.
   d. Nothing happens. The program continues running at the next statement.

Algorithm Workbench

Page 424

1. Write a statement that uses the `list` and `range` functions to create a list of the numbers from 1 to 100.

```
numbers = list(range(1,101))
```

2. Assume `names` references a list. Write a `for` loop that displays each element of the list.

```
for item in names:
    print(item)
```

OR

```
for index in range(len(names)):
    print(names[index])
```

7. What will `list1` and `list2` contain after the following statements are executed?

```
list1 = [1, 2] * 2      ——→ list1 becomes [1, 2, 1, 2]
list2 = [3]             ——→ list2 becomes [3]
list2 += list1          ——→ list2 becomes [3, 1, 2, 1, 2]
```

So after executing these statements at the end

```
list1 is [1, 2, 1, 2]
list2 is [3, 1, 2, 1, 2]
```

Checkpoint — Page 384

7.15 What is the difference between calling a list's `remove` method and using the `del` statement to remove an element?
`remove` method is used remove a specific item.
`del` statement is used to remove an item with a specific index number.

7.16 How do you find the lowest and highest values in a list?
Built-in `min` and `max` functions can be used to find the lowest and highest values in a list, respectively.

7.17 Assume the following statement appears in a program:

```
names = []
```

Which of the following statements would you use to add the string `'Wendy'` to the list at index `0`? Why would you select this statement instead of the other?

```
a. names[0] = 'Wendy'
b. names.append('Wendy')
```

When we declare/define an empty list, there is no indexing described for the list yet. So `names[0]` indexing can not be used. But `append` method can be used to add an item to the list whether if the list is empty or not. So we use b to add an item to empty list.

7.18 Describe the following list methods:
`a. index` `b. insert` `c. sort` `d. reverse`

Checkpoint — Page 421

7.26 To create a graph with the plot function, what two arguments you must pass? **X values and Y values as list**

7.27 What sort of graph does the plot function produce?

Line graphs

7.28 What functions do you use to add labels to the *X* and *Y* axes in a graph? **`xlabel and ylabel`**

7.29 How do you change the lower and upper limits of the *X* and *Y* axes in a graph? **By using `xlim and ylim functions.`**

7.30 How do you customize the tick marks along the *X* and *Y* axes in a graph? **By using `xticks and yticks functions.`**

7.31 To create a bar chart with the bar function, what two arguments you must pass? **X values and Y values (heights of the bars).**

7.32 Assume the following statement calls the bar function to construct a bar chart with four bars. What color will the bars be?

plt.bar(x_values, y_values, color=('r', 'b', 'r', 'b')) **–red-blue-red-blue**

7.33 To create a pie chart with the pie function, what argument you must pass? **values to be plotted must be passed as a list.**

Checkpoint — Page 436

8.1 Assume the variable `name` references a string. Write a `for` loop that prints each character in the string.

```
for c in name:
        print(c)
```

8.2 What is the index of the first character in a string?

Indexing goes as 0 , 1, 2, … so the first will be 0.

8.3 If a string has 10 characters, what is the index of the last character?

Indexing goes as 0 , 1, 2, … so the last will be 9 if length is 10.

8.4 What happens if you try to use an invalid index to access a character in a string?

`IndexError` exception will occur if we try to use an invalid index to access…

8.5 How do you find the length of a string?

By using `len` function. Alternatively, one can count the characters in a string as going through it b using for repetition.

8.6 What is wrong with the following code?

```
animal = 'Tiger'
animal[0] = 'L'
```

strings are immutable, using such type of statement will raise an exception error.

Checkpoint — Page 452

8.11 Write code using the `in` operator that determines whether 'd' is in `mystring`.

```
if 'd' in mystring:
```

8.12 Assume the variable `big` references a string. Write a statement that converts the string it references to lowercase and assigns the converted string to the variable `little`.

```
little=big.lower()
```

8.13 Write an if statement that displays “Digit” if the string referenced by the variable `ch` contains a numeric digit. Otherwise, it should display “No digit.”

```
 if ch.isdigit():
        print('Digit')
else:
        print('No digit')
```

8.14 What is the output of the following code?

```
ch = 'a'
ch2 = ch.upper()
print(ch, ch2)          a A
```

8.15 Write a loop that asks the user “Do you want to repeat the program or quit? (R/Q)”. The loop should repeat until the user has entered an R or Q (either uppercase or lowercase).

```
answer='R'
while answer.upper == 'R':
        …Program will be here…
        answer=input('Do you want to repeat the program or quit? (R/Q)')
```

Checkpoint

Page 452 - 453

8.16 What will the following code display?

```
var = '$'
print(var.upper())
```

it displays $ because there is no upper form of symbols.

8.17 Write a loop that counts the number of uppercase characters that appear in the string referenced by the variable `mystring`.

```
count=0
for ch in mystring:
        if ch.isupper():
                count+=1
```

8.18 Assume the following statement appears in a program:

```
days = 'Monday Tuesday Wednesday'
```

Write a statement that splits the string, creating the following list:

```
['Monday', 'Tuesday', 'Wednesday']
days_list = days.split()
```

8.19 Assume the following statement appears in a program:

```
values = 'one$two$three$four'
```

Write a statement that splits the string, creating the following list:

```
['one', 'two', 'three', 'four']
values_list = values.split('$')
```

Review Questions

Page 453-454

Multiple Choice

2. This is the last index in a string.
   a. 1
   b. 99
   c. 0
   d. The size of the string minus one
3. This will happen if you try to use an index that is out of range for a string.
   a. A `ValueError` exception will occur.
   b. An `IndexError` exception will occur.
   c. The string will be erased and the program will continue to run.
   d. Nothing—the invalid index will be ignored.
5. This string method returns a copy of the string with all leading whitespace characters removed.
   a. `lstrip`
   b. `rstrip`
   c. `remove`
   d. `strip_leading`
6. This string method returns the lowest index in the string where a specified substring is found.
   a. `first_index_of`
   b. `locate`
   c. `find`
   d. `index_of`
7. This operator determines whether one string is contained inside another string.
   a. `contains`
   b. `is_in`
   c. `==`
   d. `in`

Algorithm Workbench

Page 455

1. Assume `choice` references a string. The following `if` statement determines whether `choice` is equal to 'Y' or 'y':

   ```
   if choice == 'Y' or choice == 'y':
   ```

   Rewrite this statement so it only makes one comparison, and does not use the `or` operator. (*Hint: use either the* `upper` *or* `lower` *methods.*)

   ```
   if choice.upper() == 'Y':
   ```

2. Write a loop that counts the number of space characters that appear in the string referenced by `mystring`.

   ```
   count=0
   for c in mystring:
       if c.isspace():
           count += 1
   ```

5. Write a function that accepts a string as an argument and returns `true` if the argument starts with the substring `'https'`. Otherwise, the function should return `false`.

   ```
   def check( str ):
       return str.startswith('https')
   ```

7. Write a function that accepts a string as an argument and displays the string backwards.

   ```
   def reverse_string( str ):
       for i in range(len(str)-1,-1,-1):
           print(str[i],end='')
   ```

Checkpoint

Page 484

9.1 An element in a dictionary has two parts. What are they called?

key and value

9.2 Which part of a dictionary element must be immutable?

key part of a dictionary is immutable

9.3 Suppose 'start' : 1472 is an element in a dictionary. What is the key? What is the value?

key is 'start' and value is 1472

9.4 Suppose a dictionary named employee has been created. What does the following statement do?

```
employee['id'] = 54321
```

It modifies the id as 54321 or adds it to the employee dictionary if it is not already exist.

9.5 What will the following code display?

```
stuff = {1 : 'aaa', 2 : 'bbb', 3 : 'ccc'}
print(stuff[3])
```

It displays `ccc`

9.6 How can you determine whether a key-value pair exists in a dictionary?

by using `in` or `not in` operators.

Checkpoint

Page 484

9.7 Suppose a dictionary named inventory exists. What does the following statement do?

```
del inventory[654]
```

It deletes/removes the element with key 654 from the inventory dictionary.

9.8 What will the following code display?

```
stuff = {1 : 'aaa', 2 : 'bbb', 3 : 'ccc'}
print(len(stuff))
```

It displays 3 which is the number of elements in the stuff dictionary.

9.9 What will the following code display?

```
stuff = {1 : 'aaa', 2 : 'bbb', 3 : 'ccc'}
for k in stuff:
    print(k)
```

It displays the key values in the dictionary in separate lines.

9.10 What is the difference between the dictionary methods pop and popitem?

pop returns removes an element with a given key. popitem also does the same operation but it picks the element randomly.

9.11 What does the items method return?

Returns all the dictionaries keys and associated values as a tuple

9.12 What does the keys method return?

Returns all the dictionary keys as a sequence.

9.13 What does the values method return?

Returns all the dictionary values as a sequence.

Algorithm Workbench

Page 506

1. Write a statement that creates a dictionary containing the following key-value pairs:

```
'a' : 1
'b' : 2
'c' : 3
```

```
my_dict = {'a':1,'b':2,'c',3}
```

3. Assume the variable `dct` references a dictionary. Write an `if` statement that determines whether the key `'James'` exists in the dictionary. If so, display the value that is associated with that key. If the key is not in the dictionary, display a message indicating so.

```
if 'James' in dct:
    print(dct['James'])
else:
    print('James not found!)
```

4. Assume the variable `dct` references a dictionary. Write an `if` statement that determines whether the key `'Jon'` exists in the dictionary. If so, assign the value of `'Jon'` to a key of `'John'`, and then delete `'Jon'` and its associated value.

```
if 'Jon' in dct:
    dct['John'] = dct['Jon']
    del dtc['Jon']
```

Review Questions

Page 502-503

Multiple Choice

1. You can use the _____ operator to determine whether a key exists in a dictionary.
   a. &
   b. in
   c. ^
   d. ?
2. You use _____ to delete an element from a dictionary.
   a. the `remove` method
   b. the `erase` method
   c. the `delete` method
   d. the `del` statement
3. The _____ function returns the number of elements in a dictionary:
   a. `size()`
   b. `len()`
   c. `elements()`
   d. `count()`
4. You can use _____ to create an empty dictionary.
   a. `{}`
   b. `()`
   c. `[]`
   d. `empty()`
5. The _____ method returns a randomly selected key-value pair from a dictionary.
   a. `pop()`
   b. `random()`
   c. `popitem()`
   d. `rand_pop()`

Checkpoint

Page 495

9.15 Does a set allow you to store duplicate elements?

No, sets don't allow us to store duplicate elements. Elements must be different.

9.16 How do you create an empty set?

`set()` creates an empty set

9.17 After the following statement executes, what elements will be stored in the

```
myset set?
myset = set('Jupiter')
```

`{'r', 'i', 'e', 'J', 't', 'p', 'u'}` – Characters in Jupiter!

9.18 After the following statement executes, what elements will be stored in the `myset` set?

```
myset = set(25)
```

This is not possible in Python 3! Only iterable objects can be stored, this is only an integer value.

9.19 After the following statement executes, what elements will be stored in the `myset` set?

```
myset = set('www xxx yyy zzz')
```

`{'w', ' ', 'z', 'x', 'y'}`

9.21 After the following statement executes, what elements will be stored in the `myset` set?

```
myset = set(['www', 'xxx', 'yyy', 'zzz'])
```

`{'www', 'zzz', 'yyy', 'xxx'}` – each string is stored!

Checkpoint — Page 496

9.27 After the following code executes, what elements will be members of set3?

```
set1 = set([10, 20, 30])
set2 = set([100, 200, 300])
set3 = set1.union(set2)
```

```
{20, 100, 200, 10, 300, 30}
```

9.28 After the following code executes, what elements will be members of set3?

```
set1 = set([1, 2, 3, 4])
set2 = set([3, 4, 5, 6])
set3 = set1.intersection(set2)
```

```
{3, 4}
```

9.31 After the following code executes, what elements will be members of set3?

```
set1 = set(['a', 'b', 'c'])
set2 = set(['b', 'c', 'd'])
set3 = set1.symmetric_difference(set2)
```

```
{'d', 'a'}
```

9.32 Look at the following code:

```
set1 = set([1, 2, 3, 4])
set2 = set([2, 3])
```

Which of the sets is a subset of the other?
Which of the sets is a superset of the other?
set1 is superset to set2 set1 <= set2
set2 is subset to set1 set2 >= set1

Checkpoint — Page 502

9.33 What is object serialization?
Convert an object to a stream of bytes that can easily be stored in a file

9.34 When you open a file for the purpose of saving a pickled object to it, what file access mode do you use?
`wb` – Binary Write Mode

9.35 When you open a file for the purpose of retrieving a pickled object from it, what file access mode do you use?
`rb` – Binary Read Mode

9.36 What module do you import if you want to pickle objects?
`pickle` module is needed to be imported.

9.37 What function do you call to pickle an object?
`pickle.dump` function is used to pickle an object to a file

9.38 What function do you call to retrieve and unpickle an object?
`pickle.load` function is used to unpickle an object from a file

Review Questions

Page 503-504

Multiple Choice

11. You can add a group of elements to a set with this method.
    a. `append`
    b. `add`
    c. `update`
    d. `merge`

13. This `set` method removes an element and raises an exception if the element is not found.
    a. `remove`
    b. `discard`
    c. `delete`
    d. `erase`

15. This operator can be used to find the difference of two sets.
    a. `|`
    b. `&`
    c. `-`
    d. `^`

17. This operator can be used to find the symmetric difference of two sets.
    a. `|`
    b. `&`
    c. `-`
    d. `^`

Algorithm Workbench

Page 506-507

6. Assume each of the variables `set1` and `set2` references a set. Write code that creates another set containing all the elements of `set1` and `set2`, and assigns the resulting set to the variable `set3`.

```
set3 = set1.union(set2)  or set3 = set1 | set2
```

9. Assume each of the variables `set1` and `set2` references a set. Write code that creates another set containing the elements that appear in `set2` but not in `set1`, and assigns the resulting set to the variable `set3`.

```
set3 = set2.difference(set1)  or set3 = set2 - set1
```

10. Assume that `set1` references a set of integers and `set2` references an empty set. Write code that iterates through each element of `set1`. If the element is greater than 100, add it to `set2`.

```
for item in set1:
    if item > 100:
        set2.add(item)
```

11. Assume the variable `dct` references a dictionary. Write code that pickles the dictionary and saves it to a file named `mydata.dat`.

```
output = open( 'mydata.dat','wb' )
pickle.dump(dct, output)
```