### EXPOSIFE ES 2022 for ES1

## Chanter 17 Slides

Sequential Text Files

PowerPoint Presentation created by:
Mr. John L. M. Schram and Mr. Leon Schram Authors of Exposure computer Science



#### Section 17.1

# Introduction

#### **Programs and Data Files**

Data information processed by word processing, spreadsheet and similar application programs are stored as data files.

Python programs, written with a text editor or some Integrated Development Environment (*IDE*), are also data files.

Programs have the capability to create and retrieve their own data files. This means that data entered in any one of your programs can be saved for later use.

	ata Organization
hit	This is the fundamental building block of all comput

field

file

record

information – a binary digit – which stores only a 1 or 0.

One byte equals 8 bits. byte The ASCII format uses 1 byte to store a character. UTF-8 uses between 1 and 4 bytes to store a character.

> A *field* is one specific unit of information. Examples: name, address, gpa, salary, diagnosis

A record consists of a set of fields for a specific purpose. Examples of the fields contained by different records: student record: name, address, gpa, classRank employee record: name, address, salary, position patient record: name, address, diagnosis, allergies

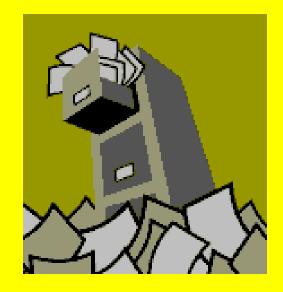
A data base file consists of a sequence of records. Example: A school can have a file of student records.

## Section 17.2 nifferent Types

#### File Data Structure Definition Review

A *file* is an *internal* data structure – with an unspecified number of elements of the same type – assigned to an *external* file name.

The file data structure allows transfer of data between internal and external storage.

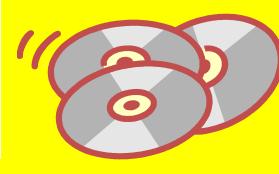


#### External File Definition

An external file is a sequence of information stored as bytes on a hard drive, disk, tape, CD, DVD, jump drive, or some other external storage device.









#### Text Files and Binary Files

Text files contain only characters and can be edited by any text editor like **Notepad** or **jGRASP**.

All Python programs are text files.

Binary files have a specific format which requires a specific application for editing.

#### 2 Ways to Classify Files

by what they <u>store</u> (text vs. binary)

 by how the information is <u>accessed</u> (sequential access vs. random access)

## Sequential Access and Random Access

Files can have sequential access or random access.

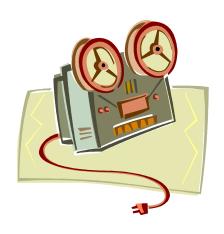
Sequential access files allow data access only in the sequence that the data is stored in the file.

Random access files allow data access in any random pattern, regardless of how the data is stored.

#### Sequential Access Examples







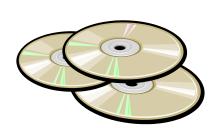






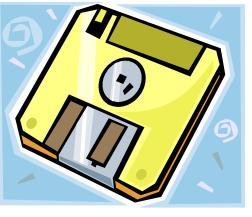
#### Random Access Examples







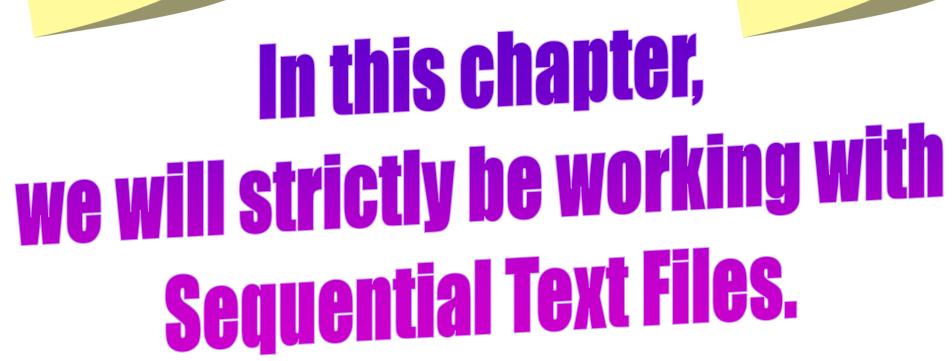








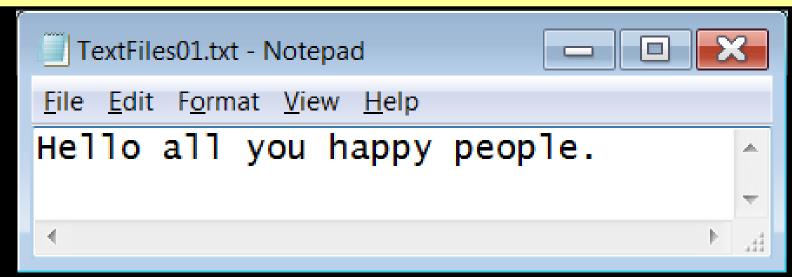
#### NOTE:



## Section 17.3 Writing To and Reading From Text FIGS

```
1 # TextFiles01.py
 2 # This program demonstrates how to create a
 3 # new text file and <write> some text to it.
 4
 5
  file = open("TextFiles01.txt",'w')
  file.write("Hello all you happy people.")
10 file.close()
```

```
1 # TextFiles01.py
 2 # This program demonstrates how to create a
 3 # new text file and <write> some text to it.
 4
 5
  file = open("TextFiles01.txt",'w')
  file.write("Hello all you happy people.")
 9
10 file.close()
```



## committer 5 mg that thou open, ou shalt also close!

```
1 # TextFiles02.py
 2 # This program demonstrates how to <read> the text
 3 # from the file created by the previous program.
  # NOTE: If the previous program, TextFiles01.py,
 5 # has not been executed, this program will crash.
 6
 8 file = open("TextFiles01.txt",'r')
10 text = file.readline()
11
12 print()
                    TextFiles01.txt - Notepad
13 print(text)
                   <u>File Edit Format View Help</u>
                   Hello all you happy people.
14
15 file.close()
16
```

```
Hello all you happy people.
   ----jGRASP: operation complete.
  file = open("TextFiles01.txt", 'r')
10 text = file.readline()
11
12 print()
                   TextFiles01.txt - Notepad
13 print(text)
                   <u>File Edit Format View Help</u>
                  Hello all you happy people.
14
15 file.close()
16
```

----jGRASP exec: python TextFiles02

```
1 # TextFiles03.py
 2 # This program demonstrates a way to check
 3 # if a file <exists> and also a way to see
 4 # how many bytes a file contains.
 5
 6
   import os
 8
 9
10 print()
11 if os.path.exists("TextFiles01.txt"):
      print("TextFiles01.txt stores",
12
            os.path.getsize("TextFiles01.txt"),
13
             'bytes of data.")
14
15 else:
      print("TextFiles01.txt does not exist.")
16
17
18 print()
   if os.path.exists("qwerty.txt"):
19
      print("qwerty.txt stores",
20
            os.path.getsize("qwerty.txt"),
21
            "bytes of data.")
22
23 else:
      print("qwerty.txt does not exist.")
24
```

```
----jGRASP exec: python TextFiles03.py
  TextFiles01.txt stores 27 bytes of data.
  qwerty.txt does not exist.
  ----jGRASP: operation complete.
  if os.path.exists("TextFiles01.txt"):
      print("TextFiles01.txt stores",
12
            os.path.getsize("TextFiles01.txt"),
13
            'bytes of data.")
14
15
  else:
      print("TextFiles01.txt does not exist.")
16
17
18
   print()
   if os.path.exists("qwerty.txt"):
19
      print("qwerty.txt stores",
20
            os.path.getsize("qwerty.txt"),
21
            "bytes of data.")
22
23 else:
      print("qwerty.txt does not exist.")
24
```

```
1 # TextFiles04.py
 2 # This program attempts to send multiple lines
 3 # of output to a text file. After you run this
 4 # program, load the file TextFiles04.txt and
 5 # you will see this did not work properly.
 6
  file = open("TextFiles04.txt",'w')
10 file.write("Hello")
11 file.write("all")
12 file.write("you")
13 file.write("happy")
14 file.write("people.")
15
16 file.close()
```

```
TextFiles04.txt - Notepad
     Edit F<u>o</u>rmat <u>V</u>iew
 File
                        Help
 Helloallyouhappypeople.
  file = open("TextFiles04.txt",'w')
 9
10 file.write("Hello")
11 file.write("all")
12 file.write("you")
13 file.write("happy")
14 file.write("people.")
15
16 file.close()
```

```
1 # TextFiles05.py
 2 # This program fixes the issue of the
 3 # previous program by adding the
 4 # "new line escape sequence" <\n>.
 6
 7 file = open("TextFiles05.txt",'w')
 9 file.write("Hello\n")
10 file.write("all\n")
11 file.write("you\n")
12 file.write("happy\n")
13 file.write("people.\n")
14
15 file.close()
```

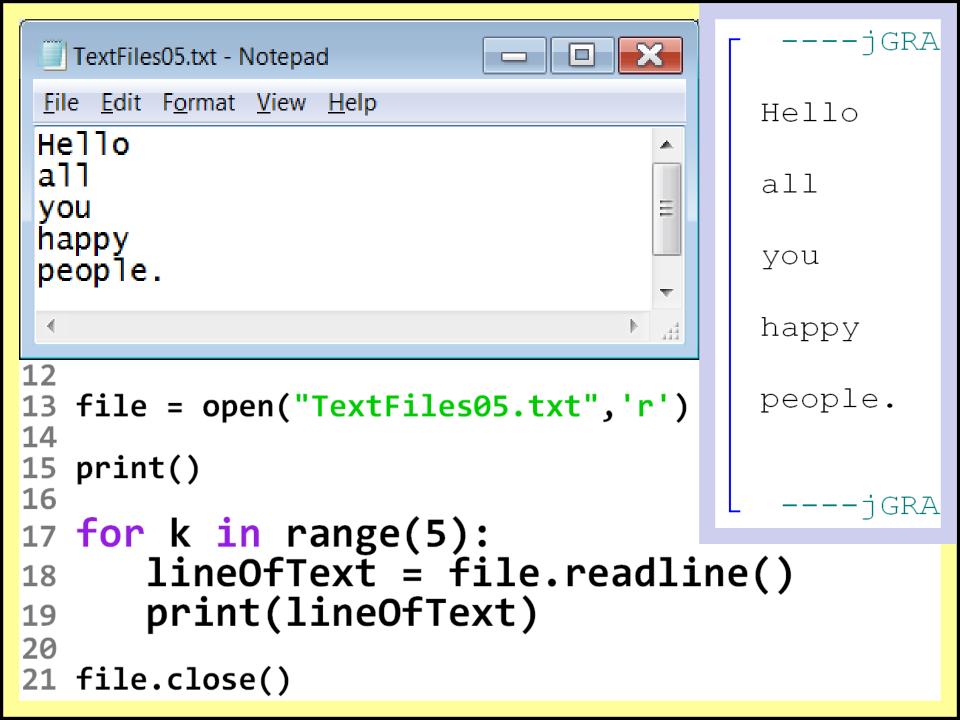
```
1 # TextFiles05.py
 2 # This program fixes the issue of the
     previous program by adding the
 4 # "new line escape sequence" <\n>.
 6
 7 file = open("TextFiles05.txt",'w')
                                    TextFiles05.txt -
  file.write("Hello\n")
                                  <u>F</u>ile <u>E</u>dit
                                         Format
10 file.write("all\n")
                                  Hello
11 file.write("you\n")
                                  аПТ
12 file.write("happy\n")
                                  you
                                  happy
13 file.write("people.\n")
                                  people.
14
15 file.close()
```

```
1 # TextFiles06.py
 2 # This program tries to read the file created
 3 # by the previous program. The problem is only
  # one <readline> command is executed, so only
 5 # one line of text is read and displayed.
  # NOTE: If the previous program, TextFiles05.py,
           has not been executed, this program, and
 7 #
8 #
           the next several programs, will crash.
10
                                           TextFiles05.txt -
11 file = open("TextFiles05.txt",'r')
                                         <u>File</u> <u>E</u>dit
                                                 Format
12
                                        Hello
13 lineOfText = file.readline()
                                        аПТ
14
                                        you
15 print()
                                        happy
16 print(lineOfText)
                                        people.
17
18 file.close()
```

```
----jGRASP exec: python TextFiles06.py
  Hello
    ----jGRASP: operation complete.
 9
10
                                           TextFiles05.txt -
11 file = open("TextFiles05.txt",'r')
                                                 Format
                                         <u>File</u> <u>E</u>dit
12
                                         Hello
13 lineOfText = file.readline()
14
                                         you
15 print()
                                        happy
16 print(lineOfText)
                                        people.
18 file.close()
```

```
1 # TextFiles07.py
 2 # This program uses a <for> loop to read
 3 # all 5 lines of text from the file.
 4 # There are a couple problems. First, there
 5 # are extra lines skipped in the output, caused
 6 # by the <\n> characters used to write the
 7 # data when the file was created.
8 # Second, this program only works because we
9 # know, ahead of time, how many lines of text
10 # are in the file, which is not realistic.
11
12
13 file = open("TextFiles05.txt", 'r')
14
15 print()
16
17 for k in range(5):
      lineOfText = file.readline()
18
       print(lineOfText)
19
20
21 file.close()
```

```
TextFiles05.txt - Notepad
                                         read
 File Edit Format View Help
                                         st, there
 Hello
                                        ıtput, caused
 all
                                        ite the
 you
 happy
                                        ecause we
 people.
                                        nes of text
                                        ilistic.
12
  file = open("TextFiles05.txt",'r')
14
   print()
16
17 for k in range(5):
       lineOfText = file.readline()
18
       print(lineOfText)
19
20
21 file.close()
```



```
1 # TextFiles08.py
 2 # This program fixes the first issue of the
 3 # previous program by using the <strip> command
 4 # to "strip" away any invisible characters
 5 # before and after each line of text.
 6 # This includes </n> characters.
 8
  file = open("TextFiles05.txt", 'r')
10
11 print()
12
13 for k in range(5):
      lineOfText = file.readline().strip()
14
      print(lineOfText)
15
16
17 file.close()
```

```
TextFiles05.txt - Notepad
 File Edit
       Format View
                 Help
 Hello
                                            Hello
 all
 you
                                            all
 happy
 people.
                                            you
                                            happy
  file = open("TextFiles05.txt",'r')
                                            people.
10
   print()
12
13 for k in range(5):
      lineOfText = file.readline().strip()
14
      print(lineOfText)
15
16
17 file.close()
```

```
1 # TextFiles09.py
 2 # This program demonstrates what happens when you
 3 # attempt to read past the end of a text file.
 4 # In most languages, this would crash the program.
 5 # In Python, you just get extra blank lines.
 6
  file = open("TextFiles05.txt", 'r')
 9
10 print()
11
12 for k in range(8):
13
      lineOfText = file.readline().strip()
      print(lineOfText)
14
15
16 file.close()
17
```

```
TextFiles05.txt - Notepad
                                     that happe
 <u>File Edit Format View Help</u>
                                                    Hello
                                       of a te
Hello
all
                                                    all
                                      ld crash
 you
                                     ra blank
                                                    you
 happy
people.
                                                    happy
                                                    people.
   file = open("TextFiles05.txt", 'r')
 9
   print()
11
12 for k in range(8):
       lineOfText = file.readline().strip()
13
       print(lineOfText)
14
15
16 file.close()
17
```

```
1 # TextFiles10.py
 2 # This program fixes the second issue from
 3 # program TextFiles07.py by using the same
4 # <for..each> loop that works with arrays.
 5
  file = open("TextFiles05.txt", 'r')
 8
  print()
10
11 for lineOfText in file:
      lineOfText = lineOfText.strip()
12
13
      print(lineOfText)
14
15 file.close()
16
```

```
TextFiles05.txt - Notepad
                               second iss
File Edit Format View Help
Hello
                               by using t
all
                                              Hello
                              works with
you
happy
                                              all
people.
                                              you
   file = open("TextFiles05.txt",'r')
                                              happy
                                              people.
   print()
10
11 for lineOfText in file:
       lineOfText = lineOfText.strip()
12
       print(lineOfText)
13
14
15 file.close()
16
```

```
1 # TextFiles11.py
2 # This program demonstrates the <readlines> command which
3 # reads in the ENTIRE file and creates an array of strings.
 5
 6 file = open("TextFiles05.txt",'r')
7 allLinesOfText = file.readlines()
8 file.close()
10 print()
11 for lineOfText in allLinesOfText:
      lineOfText = lineOfText.strip()
12
      print(lineOfText)
13
14
```

```
- 0 X
 TextFiles05.txt - Notepad
 File Edit Format View Help
                                   <readlines> co
Hello
all
                                                      Hello
                                   creates an arra
 vou
happy
                                                      all
people.
                                                      you
 6 file = open("TextFiles05.txt",'r')
                                                      happy
                                                      people.
 7 allLinesOfText = file.readlines()
 8 file.close()
10 print()
11 for lineOfText in allLinesOfText:
      lineOfText = lineOfText.strip()
12
      print(lineOfText)
13
14
```

```
1 # TextFiles12.py
 2 # This program allows the user to enter a file name.
 3 # Then it displays the contents of that file,
  # provided it exists.
 5
 6
   import os
89
10 print()
11 fileName = input("Enter the name of a file. -->
12 print()
13
14 if os.path.exists(fileName):
     file = open(fileName, 'r')
15
16 for text in file:
        text = text.strip()
17
18
        print(text)
     file.close()
19
20 else:
     print(fileName, "does not exist.")
21
```

```
----jGRASP exec: python TextFiles12.py
Enter the name of a file. --> TextFiles01.txt
                                     TextFiles01.txt - Notepad
                                                           Hello all you happy people.
                                     <u>File Edit Format View Help</u>
 ----jGRASP: operation complete
                                     Hello all you happy people.
 ----jGRASP exec: python TextFil
Enter the name of a file. --> TextFiles05.txt
Hello
                                     TextFiles05.txt - Notepad
all
                                     <u>File Edit Format View Help</u>
you
                                     Hello
happy
                                     a11
people.
                                     vou
                                     happy
 ----jGRASP: operation complete
                                     people.
 ----jGRASP exec: python TextFil
Enter the name of a file. --> qwerty.txt
qwerty.txt does not exist.
 ----jGRASP: operation complete.
```

```
----jGRASP exec: python TextFiles12.py
Enter the name of a file. --> TextFiles12.py
# TextFiles12.py
# This program allows the user to enter a file name.
# Then it displays the contents of that file,
# provided it exists.
import os
print()
fileName = input("Enter the name of a file. --> ")
print()
if os.path.exists(fileName):
file = open(fileName,'r')
for text in file:
text = text.strip()
print(text)
file.close()
else:
print(fileName, "does not exist.")
 ----jGRASP: operation complete.
```

```
-jGRASP exec: python TextFiles12.py
Enter the name of a file. --> TextFiles12.py
# TextFiles12.py
# This program allows the user to enter a file name.
# Then it displays the contents of that file,
# provided it exists.
import os
print()
fileName = input("Enter the name of a file. --> ")
print()
if os.path.exists(fileName):
                                   Yes, this program is
file = open(fileName, 'r')
for text in file:
                                   actually displaying
text = text.strip()
print(text)
                                   itself, but what
file.close()
else:
                                   happened to the
print(fileName, "does not exist.")
                                   indenting?
```

----jGRASP: operation complete.

```
-jGRASP exec: python TextFiles12.py
Enter the name of a file. --> TextFiles12.py
# TextFiles12.py
# This program allows the user to enter a file name.
# Then it displays the contents of that file,
# provided it exists.
import os
print()
fileName = input("Enter the name of a file. --> ")
print()
if os.path.exists(fileName):
file = open(fileName,'r')
for text in file:
print(text)
                                   itself, but what
file.close()
else:
print(fileName, "does not exist.")
```

----jGRASP: operation complete.

Yes, this program is actually displaying happened to the indenting?

#### Section 17.4

# Files of Numbers

```
1 # TextFiles13.py
 2 # This program attempts to create a file of 10 random
  # integers. The program has a syntax error because
  # only string values can be written to a text file.
 5
 6
 7 from random import *
 8
  seed(1234)
10
11 file = open("TextFiles13.txt",'w')
12
13 for k in range(10):
      number = randint(1000,9999)
14
      file.write(number)
15
16
17 file.close()
18
```

```
Traceback (most recent call last):
    File "TextFiles13.py", line 15, in <module>
      file.write(number)
  TypeError: write() argument must be str, not int
   ----jGRASP wedge2: exit code for process is 1.
   ----jGRASP: operation complete.
10
11 file = open("TextFiles13.txt",'w')
12
13 for k in range(10):
      number = randint(1000,9999)
14
      file.write(number)
15
16
17 file.close()
18
```

----jGRASP exec: python TextFiles13.py

```
1 # TextFiles14.py
 2 # This program fixes the issue of the previous program
 3 # by using the <str> command to convert the numbers to
4 # strings. However, a new issue is created; which can
 5 # be seen by loading the file TextFiles15.txt
 6
  from random import *
9
10 seed(1234)
11
12 file = open("TextFiles14.txt",'w')
13
14 for k in range(10):
     number = randint(1000,9999)
15
     file.write(str(number))
16
17
18 file.close()
19
```

```
1 # TextFiles14.py
 TextFiles14.txt - Notepad
                                                 <u>File Edit Format View Help</u>
8220291411222485157223752612681048791285
  from random import
9
  seed(1234)
11
  file = open("TextFiles14.txt",'w')
13
14 for k in range(10):
      number = randint(1000,9999)
15
      file.write(str(number))
16
17
  file.close()
18
19
```

```
1 # TextFiles15.py
 2 # This program fixes the issue of the previous
 3 # program by adding a "new line escape sequence" <\n>
 4 # as was done in earlier program examples.
 5
 6
 7 from random import *
8
 9 seed(1234)
10
11 file = open("TextFiles15.txt",'w')
12
13 for k in range(10):
      number = randint(1000,9999)
14
      file.write(str(number)+"\n")
15
16
17 file.close()
18
```

```
1 # TextFiles15.py
                                          TextFiles15.txt - Notepad
 2 # This program fixes the issue of th
                                          File Edit Format View H
   # program by adding a "new line esca
                                         8220
 4 # as was done in earlier program exa
                                          2914
 5
                                          1122
 6
                                         2485
 7 from random import *
                                         1572
 8
                                         2375
  seed(1234)
                                         2612
10
                                         6810
   file = open("TextFiles15.txt",'w')
                                         4879
12
                                         1285
13 for k in range(10):
      number = randint(1000,9999)
14
      file.write(str(number)+"\n")
15
16
17 file.close()
18
```

```
1 # TextFiles16.py
 2 # This program attempts to read in the numbers from the
  # file created by the previous program and average them.
  # The problem is the numbers are stored as text and need to
  # be converted back to numbers before they can be averaged.
6
7
   file = open("TextFiles15.txt",'r')
  print()
11
12 total = 0
  count = 0
                                                          X
                                      TextFiles15.txt - Notepad
14
                                      <u>File Edit Format View Help</u>
15 for number in file:
                                     8220
  print(number)
16
                                     2914
     total += number
17
                                      1122
     count += 1
18
                                     2485
19
                                      1572
20 file.close()
                                      2375
21
                                      2612
22 average = total / count
23
                                     6810
  print()
24
                                     4879
   print("Total =",total)
                                      1285
  print()
26
   print("Average =",average)
```

```
----jGRASP exec: python TextFiles16.py
  8220
  Traceback (most recent call last):
   File "TextFiles16.py", line 17, in <module>
      total += number
  TypeError: unsupported operand type(s) for +=: 'int' and 'str'
12 total = 0
13 count = 0
                                                     TextFiles15.txt - Notepad
14
                                    <u>File Edit Format View Help</u>
15 for number in file:
                                    8220
16 print(number)
                                    2914
17 total += number
                                    1122
2485
19
                                    1572
20 file.close()
                                    2375
21
                                    2612
22 average = total / count
```

6810

4879

1285

23

24

print()

26 print()

print("Total =",total)

print("Average =",average)

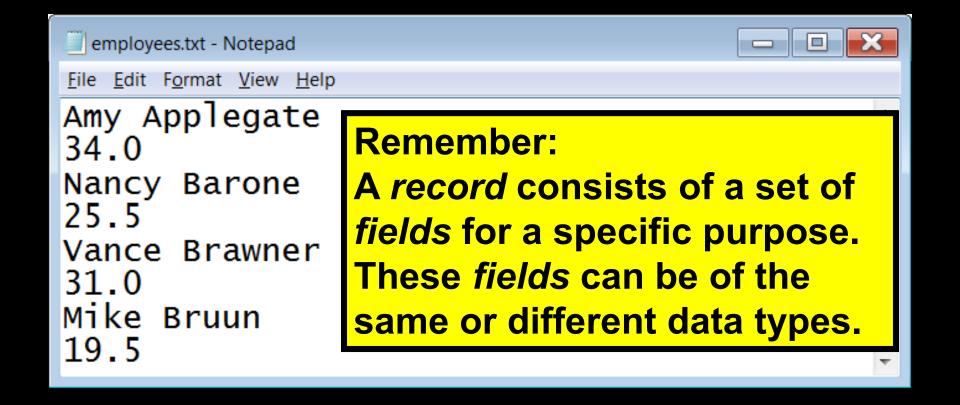
```
# TextFiles17.py
  # This program fixes the issue of the previous
  # program by using the <int> command to convert
   # the text numbers back into integer values.
  # NOTE: This process also eliminates the <\n>
           escape sequence character.
 6
7
  #
                                        TextFiles15.txt - Notepad
 9 file = open("TextFiles15.txt",'r"
                                        File Edit Format View Help
  print()
                                        8220
   total = 0
                                        2914
12 count = 0
                                        1122
13
                                        2485
14 for lineOfText in file:
                                        1572
     number = int(lineOfText)
15
                                        2375
16 print(number)
                                        2612
17 total += number
                                        6810
18
     count += 1
                                        4879
19
                                        1285
20 file.close()
21
22 average = total / count
23
24
  print()
   print("Total =",total)
   print()
26
   print("Average =",average)
```

```
1 # TextFiles17.py
 2 # This program fixes the issue of the previous
  # program by using the <int> command to convert
  # the text numbers back into integer values.
  # NOTE: This process also eliminates the <\n>
6
7
           escape sequence character.
  #
                                                ----jGRASP exec:
                                       TextFile
9 file = open("TextFiles15.txt",'r'
                                       File Edit
10 print()
                                               8220
                                      8220
  total = 0
                                               2914
                                       2914
12 count = 0
                                               1122
                                      1122
13
                                      2485
                                               2485
14 for lineOfText in file:
                                      1572
                                               1572
     number = int(lineOfText)
15
                                      2375
                                               2375
16 print(number)
                                       2612
                                               2612
17 total += number
                                      6810
                                               6810
18
   count += 1
                                      4879
                                               4879
19
                                      1285
20 file.close()
                                               1285
21
22 average = total / count
                                                       = 34274
                                               Total
23
24 print()
                                               Average = 3427.4
  print("Total =",total)
26
  print()
   print("Average =",average)
                                                ----jGRASP: oper
```

## Section 17.5 riles of Multiple

#### What Files Can Store

Files are not limited to merely storing either strings OR numbers. They can store BOTH strings AND numbers at the same time in the same file. This means that files can store *records* of information.



```
1 # TextFiles18.py
 2 # This program demonstrates how to read information
 3 # from a file containing different data types.
4 # In this case, the "employees.txt" file contains the
 5 # names (string values) and hourly rates (real# values)
6 # for several employees.
7 # The file information is read into 2 "parallel arrays".
8 # After that, the information in both arrays is displayed,
9 # and the average hourly rate is computed and displayed.
10
11
12 file = open("employees.txt", 'r')
13 names = []
14 hourlyRates = []
15 count = 0
  for lineOfText in file:
       if (count % 2 == 0):
17
           names.append(lineOfText.strip())
18
      else:
19
          hourlyRates.append(float(lineOfText))
20
       count += 1
21
22 file.close()
23
24 \text{ count} = 0
25 \text{ total} = 0
26 for k in range(len(names)):
27
     print()
     print("Employee Name: ",names[k])
28
                           ","${:.2f}".format(hourlyRates[k]))
29
     print("Hourly Rate:
     total += hourlyRates[k]
30
     count += 1
31
32
33 average = total / count
34
35 print("\n")
36 print("Average Hourly Rate:","${:.2f}".format(average))
```

```
1 # TextFiles18.py
                                                                          ----jGRASP exec: python TextFil
 2 # This program demonstrates how to read information
 3 # from a file containing different data types.
                                                                         Employee Name: Amy Applegate
 4 # In this case, the "employees.txt" file contains the
                                                                         Hourly Rate:
                                                                                       $34.00
 5 # names (string values) and hourly rates (real# values)
 6 # for several employees.
                                                                         Employee Name: Nancy Barone
 7 # The file information is read into 2 "parallel arrays".
                                                                         Hourly Rate:
                                                                                       $25.50
 8 # After that, the information in both arrays is displayed,
 9 # and the average hourly rate is computed and displayed.
                                                                         Employee Name: Vance Brawner
10
                                                                         Hourly Rate:
                                                                                       $31.00
11
12 file = open("employees.txt",'r')
                                                                         Employee Name: Mike Bruun
13 names = []
                                                                         Hourly Rate:
                                                                                       $19.50
14 hourlyRates = []
                                                                         Employee Name: Gordon Collins
15 count = 0
                                                                         Hourly Rate:
                                                                                       $25.00
  for lineOfText in file:
       if (count % 2 == 0):
17
            names.append(lineOfText.strip())
18
       else:
                                                                         Employee Name:
                                                                                       Chris Stark
19
                                                                         Hourly Rate:
                                                                                       $80.00
            hourlyRates.append(float(lineOfText))
20
       count += 1
21
                                                                         Employee Name: Tom Tooch
22 file.close()
                                                                         Hourly Rate:
                                                                                       $26.50
23
24 \text{ count} = 0
                                                                         Employee Name: Michael Ward
25 \text{ total} = 0
                                                                         Hourly Rate:
                                                                                       $19.00
26 for k in range(len(names)):
27
      print()
                                                                         Employee Name: Cheryl Willis
      print("Employee Name: ",names[k])
print("Hourly Rate: ","${:.2f}"
28
                                                                         Hourly Rate:
                                                                                       $37.00
                              ","${:.2f}".format(hourlyRates[k]))
29
30
      total += hourlyRates[k]
                                                                                       Ziggy Zighlander
                                                                         Employee Name:
      count += 1
31
                                                                         Hourly Rate:
                                                                                       $77.50
32
33 average = total / count
34
                                                                         Average Hourly Rate: $34.43
35 print("\n")
36 print("Average Hourly Rate:","${:.2f}".format(average))
                                                                          ----jGRASP: operation complete.
```

### Parallel Arrays Example

names		
0	Amy Applegate	
1	Nancy Barone	
2	Vance Brawner	
3	Mike Bruun	
4	Gordon Collions	
:	:	
35	Ziggy Zighlander	

hourlyRates	
0	34.0
1	25.5
2	31.0
3	19.5
4	25.0
:	<b>:</b>
35	77.5



### Parallel Arrays Disclaimer



When working with multiple records of information, the use of *Parallel Arrays* is not the only option, nor is it the best option.

It is much more organized, efficient, and even intuitive to use an array of records to store all of the information.

However, this requires a complexity in coding that we wish to avoid in a first year computer science class.

Arrays of Records will be discussed along with Object Oriented Programming in AP® Computer Science-A.

## Section 17.6 Reading & Writing simultaneously

```
1 # TextFiles19.py
 2 # This program reads an original file and writes
 3 # a backup file with the exact same contents.
 4 # It demonstrates that a program can read from
 5 # and/or write to multiple files simultaneously.
 6 # After you run the program load both original.txt
7 # and backup.txt and you should see the two files
 8 # are identical.
10
11 inFile = open("original.txt",'r')
12 outFile = open("backup.txt",'w')
13
14 print()
15
16 for lineOfText in inFile:
      outFile.write(lineOfText)
17
18
19 inFile.close()
20 outFile.close()
```

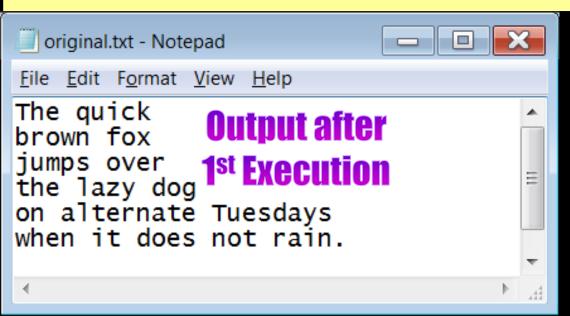
```
1 # TextFiles19.py
                                        original.txt - Notepad
 2 # This program reads an original
 3 # a backup file with the exact san
                                        <u>File Edit Format View</u>
 4 # It demonstrates that a program
                                        The quick
 5 # and/or write to multiple files
                                        brown fox
 6 # After you run the program load
                                        jumps over
 7 # and backup.txt and you should se
                                        the lazy dog
 8 # are identical.
10
11 inFile = open("original.txt",'r')
12 outFile = open("backup.txt",'w')
13
14 print()
15
16 for lineOfText in inFile:
      outFile.write(lineOfText)
17
18
19 inFile.close()
20 outFile.close()
```

```
1 # TextFiles19.py
                                         original.txt - Notepad
 2 # This program reads an original
 3 # a backup file with the exact san
                                         File Edit Format View
 4 # It demonstrates that a program
                                        The quick
 5 # and/or write to multiple files
                                        brown fox
 6 # After you run the program load
                                        jumps over
 7 # and backup.txt and you should se
                                        the lazy dog
 8 # are identical.
10
11 inFile = open("original.txt",'r')
                                         backup.txt - Notepad
12 outFile = open("backup.txt",'w')
13
                                         <u>File Edit Format View</u>
14 print()
                                        The quick
15
                                        brown fox
16 for lineOfText in inFile:
                                         jumps over
      outFile.write(lineOfText)
17
                                        the lazy dog
18
19 inFile.close()
20 outFile.close()
```

## Section 17.7 Anneming to an Existing File

```
1 # TextFiles20.py
2 # This program demonstrates how to "append"
3 # data to the end of an existing file.
 4
  file = open("original.txt", 'a')
  file.write("on alternate Tuesdays\n")
  file.write("when it does not rain.\n")
10
11 file.close()
```

```
1 # TextFiles20.py
2 # This program demonstrates how to "append"
3 # data to the end of an existing file.
 5
  file = open("original.txt", 'a')
  file.write("on alternate Tuesdays\n")
  file.write("when it does not rain.\n")
10
11 file.close()
```



```
1 # TextFiles20.py
 2 # This program demonstrates how to "append"
 3 # data to the end of an existing file.
    file = open("original.txt", 'a')
    file.write("on alternate Tuesdays\n")
   file.write("when it does not rain.\n")
10
11 file.close()
                          original.txt - Notepad
                                                 <u>File Edit Format View Help</u>
original.txt - Notepad
                         The quick
File Edit Format View Help
                                          Output after
                         brown fox
The quick
            Output after
                         jumps over
                                         2<sup>nd</sup> Execution
brown fox
                         the lazy dog
the lazy dog
                         on alternate Tuesdays
                         when it does not rain.
on alternate Tuesdays
                         on alternate Tuesdays
when it does not rain.
                         when it does not rain.
```

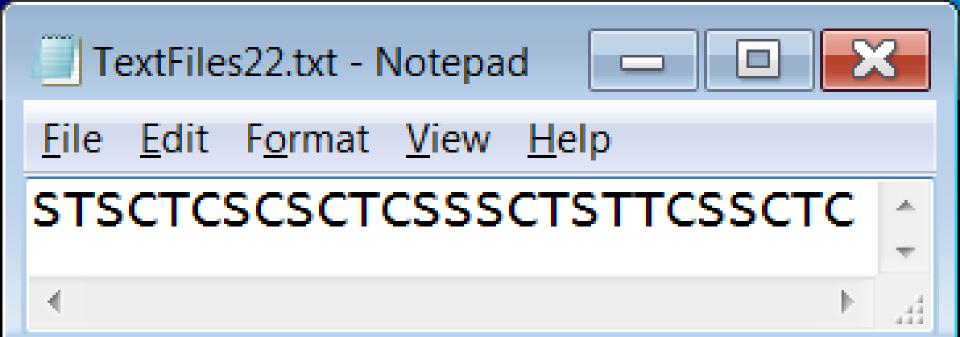
## Section 17.8 Using Text File Data in Granhics Programs

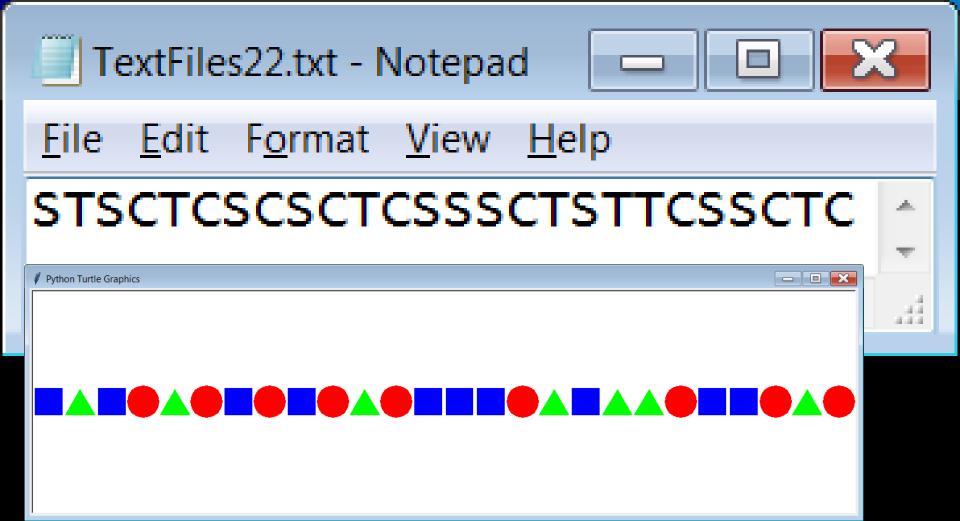
```
TextFiles21.py
     This program demonstrates that data from a
 3 # text file can be used in a graphics program.
  from Graphics import *
 9 file = open("TextFiles12.txt",'r')
10
11 x1 = int(file.readline())
12 y1 = int(file.readline())
                                    TextFiles21.txt - Notepad
13 r1 = int(file.readline()
14 x2 = int(file.readline()
                                  <u>File Edit Format View Help</u>
15 y2 = int(file.readline(
                                  500
16 r2 = int(file.readline()
                                  325
17 x3 = int(file.readline()
18 y3 = int(file.readline()
                                  200
19 r3 = int(file.readline()
                                  400
20 x4 = int(file.readline()
                                  250
21 y4 = int(file.readline(
22 r4 = int(file.readline()
                                  50
23 x5 = int(file.readline()
                                  600
24 y5 = int(file.readline()
                                  250
25 r5 = int(file.readline()
                                  50
26 x6 = int(file.readline()
27 y6 = int(file.readline()
                                  416
28 r6 = int(file.readline()
                                  270
29 r7 = int(file.readline())
                                  25
30
                                  585
31 file.close()
                                  270
32
33 beginGrfx(1000,650)
                                  25
34
                                  500
35 drawCircle(x1,y1,r1)
                                  400
36 drawCircle(x2,y2,r2)
                                  150
37 drawCircle(x3,y3,r3)
38 fillCircle(x4,y4,r4)
                                  50
39 fillCircle(x5,y5,r5)
40 draw0val(x6,y6,r6,r7)
41
42 endGrfx()
```

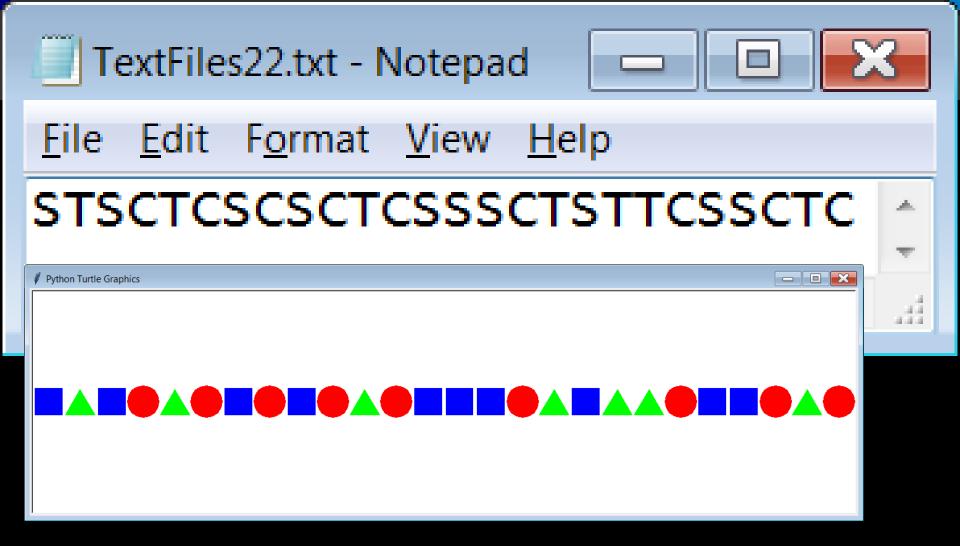
```
TextFiles21.py
     This program demonstrates that data from a
 3 # text file can be used in a graphics program.
  from Graphics import *
 9 file = open("TextFiles12.txt",'r')
10
11 x1 = int(file.readline())
12 y1 = int(file.readline())
                                     TextFiles21.txt - Notepad
13 r1 = int(file.readline()
14 x2 = int(file.readline()
                                      <u>E</u>dit F<u>o</u>rmat <u>V</u>iew <u>H</u>elp
                                  <u>F</u>ile
15 y2 = int(file.readline(
                                  500
16 r2 = int(file.readline()
                                         17 x3 = int(file.readline()
                                  325
18 y3 = int(file.readline(
                                  200
19 r3 = int(file.readline()
                                  400
20 x4 = int(file.readline()
                                  250
21 y4 = int(file.readline(
22 r4 = int(file.readline()
                                  50
23 x5 = int(file.readline()
                                  600
24 y5 = int(file.readline()
                                  250
25 r5 = int(file.readline()
                                  50
26 x6 = int(file.readline()
27 y6 = int(file.readline()
                                  416
28 r6 = int(file.readline()
                                  270
29 r7 = int(file.readline())
                                  25
30
                                  585
31
  file.close()
                                  270
32
33 beginGrfx(1000,650)
                                  25
34
                                  500
35 drawCircle(x1,y1,r1)
                                  400
36 drawCircle(x2,y2,r2)
                                  150
37 drawCircle(x3,y3,r3)
38 fillCircle(x4,y4,r4)
                                  50
39 fillCircle(x5,y5,r5)
40 draw0val(x6,y6,r6,r7)
41
42 endGrfx()
```

```
1 # TextFiles22.py
 2 # This program demonstrates that non-numerical
 3 # data can also be used in a graphics program.
 4
 5
  from Graphics import *
 7
8
   def drawBlueSquare(x):
      setColor("blue")
10
      x = x * 50 + 25
11
      fillRegularPolygon(x, 175, 30, 4)
12
13
14
15 def drawRedCircle(x):
      setColor("red")
16
      x = x * 50 + 25
17
      fillCircle(x, 175, 25)
18
19
20
   def drawGreenTriangle(x):
21
      setColor("green")
22
      x = x * 50 + 25
23
      fillRegularPolygon(x,183,27,3)
24
```

```
25
26
27
   #########
28
29
   #
      MATN
30
   ##########
31
32 file = open("TextFiles22.txt",'r')
33 lineOfText = file.readline()
34 file.close()
35
   beginGrfx(1300,350)
37
38 for x in range(len(lineOfText)):
      if lineOfText[x] == "S":
39
         drawBlueSquare(x)
40
      elif lineOfText[x] == "C":
41
42
         drawRedCircle(x)
      elif lineOfText[x] == "T":
43
         drawGreenTriangle(x)
44
45
   endGrfx()
46
47
48
```



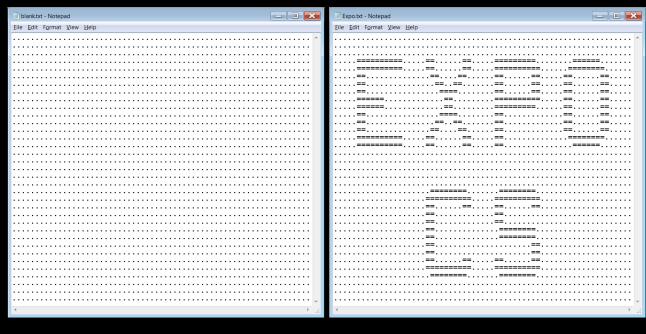


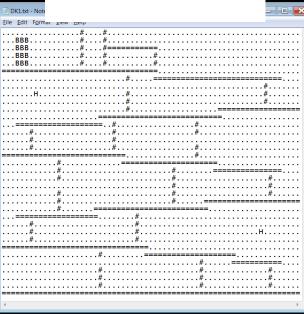


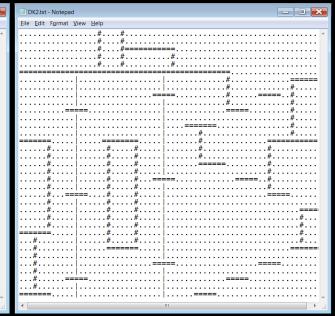
Note how the graphics output precisely correlates to the characters in the text file. Try altering the characters in this file and run the program again to see what happens.

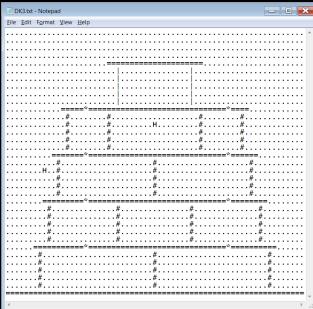
## TextFiles23.py Data Files











```
1 # TextFiles23.py
 2 # This program will ask the user to enter
 3 # the name of a text file, and then use that
4 # text file to display a graphics background.
 5 # Students are provided with 5 example files:
 6 # blank.txt, Expo.txt, DK1.txt, DK2.txt, and
7 # DK3.txt -- the latter 3 resemble stages from
 8 # Nintendo's Donkey Kong arcade game.
9 # NOTE: This same program is used for Lab 16B.
10
11
12 from Graphics import *
13 import os
14
15
16 def convert(q):
17
      return q * 20
18
19
20 def drawSpace(r,c):
21
      x = convert(c)
22
      y = convert(r)
23
      setColor("black")
24
      fillRect(x,y,20,20)
25
26
27 def drawGirder(r,c):
28
      x = convert(c)
29
      v = convert(r)
      setColor("red");
30
31
      fillRect(x,y,20,20)
32
      setColor("black")
33
      fillCircle(x+10,y+9,6)
34
35
```

```
36 def drawLadder(r,c):
37
      x = convert(c)
38
      v = convert(r)
      setColor("black")
39
40
      fillRect(x,y,20,20)
41
      setColor("white")
      fillRect(x,y,3,20)
42
43
      fillRect(x+16,y,4,20)
44
      fillRect(x,y+8,20,4)
45
46
47 def drawHammer(r,c):
48
      x = convert(c)
49
      v = convert(r)
50
      setColor("black")
      fillRect(x,y,20,20)
51
52
      setColor(150,100,15)
      fillRect(x,y,20,10)
53
54
      setColor("yellow");
55
      fillRect(x+8,y+10,4,10)
56
57
58 def drawBarrel(r,c):
59
      x = convert(c)
60
      v = convert(r)
61
      setColor("black")
      fillRect(x,y,20,20)
62
63
      setColor(150,100,15)
64
      fillRect(x+5,y,10,20)
65
      fillArc(x+5,y+10,5,10,180,360)
66
      fillArc(x+15,y+10,5,10,0,180)
67
      setColor("black")
68
      drawLine(x+16,y+9,x+16,y+13)
69
      drawLine(x+15,y+6,x+15,y+16)
70
      setColor("white")
71
      drawLine(x+5,y+4,x+5,y+14)
72
      drawLine(x+4,y+7,x+4,y+10)
73
      setColor(211,211,211)
74
      drawLine(x+5,y,x+15,y)
```

```
75
                                        105 ##########
                                        106 # MAIN
 76
                                        107 ##########
77 def drawLock(r,c):
                                        108
 78
       x = convert(c)
                                        109 \text{ numRows} = 35
 79
                                        110 \text{ numCols} = 65
       y = convert(r)
                                        111 fileName = textinput("Graphics Background",
 80
       setColor("cyan")
                                        "Enter the name of the text file.")
 81
       fillRect(x,y,20,5)
                                        112 while not os.path.exists(fileName):
 82
       setColor("yellow")
                                                fileName = textinput("File does not exist.",
                                        113
 83
       fillRect(x,y+5,20,15)
                                        "Enter the name of the text file.")
                                        114 file = open(fileName, "r")
 84
                                        115 background = file.readlines()
 85
                                        116 file.close()
 86 def drawPole(r,c):
                                        117
 87
       x = convert(c)
                                        118 beginGrfx(1300,700)
                                        119
 88
       y = convert(r)
                                        120 for r in range(numRows):
 89
       setColor("black")
                                        121
                                                for c in range(numCols):
 90
       fillRect(x,y,20,20)
                                        122
                                                   if background[r][c] == '.':
       setColor("cyan")
 91
                                        123
                                                      drawSpace(r,c)
       fillRect(x+7,y,6,20)
 92
                                        124
                                                   elif background[r][c] == '=':
                                        125
                                                      drawGirder(r,c)
 93
                                        126
                                                   elif background[r][c] == '#':
 94
                                        127
                                                      drawLadder(r,c)
 95 def drawUnknown(r,c):
                                        128
                                                   elif background[r][c] == 'H':
 96
       x = convert(c)
                                        129
                                                      drawHammer(r,c)
                                        130
                                                   elif background[r][c] == 'B':
 97
       v = convert(r)
                                        131
                                                      drawBarrel(r,c)
 98
       setColor("pink")
                                        132
                                                   elif background[r][c] == '*':
 99
       fillRect(x,y,20,20)
                                        133
                                                      drawLock(r,c)
100
       setColor("black")
                                        134
                                                   elif background[r][c] == '|':
                                        135
101
       drawString("?",x+3,y+25,
                                                      drawPole(r,c)
                                        136
                                                   else:
"Courier", 16, "bold")
                                        137
                                                      drawUnknown(r,c)
102
                                        138
                                                update()
103
                                        139
104
                                        140 endGrfx()
```

