Introduction to Programming

Department of Computer Science and Information Systems

Lecturers: Tingting Han and Steve Maybank

sjmaybank@dcs.bbk.ac.uk

Autumn 2019 and Spring 2020

Week 8: Loops

Lab 7, Quiz Grading

Score	Grade
90-100	A
80-89	B
70-79	C
60-69	D
<60	E

Obtain an integer valued score from the keyboard and print out the corresponding letter grade.

Solution to Quiz Grading

```
score = int(input("Enter the score: "))
grade = ""
if (score >= 90) :      #brackets are not essential
    grade = "A"
elif (score >= 80) :
    grade = "B"
elif (score >= 70) :
    grade = "C"
elif (score >= 60) :
    grade = "D"
else :                  #no elif is allowed after else
    grade = "E"
```

Score	Grade
90-100	A
80-89	B
70-79	C
60-69	D
<60	E

Solution to Quiz Grading

```
score = int(input("Enter the score: "))
grade = ""
```

```
if (score >= 90) :
    grade = "A"
elif (score >= 80) :
    grade = "B"
elif (score >= 70) :
    grade = "C"
elif (score >= 60) :
    grade = "D"
else :
    grade = "E"
```

```
if (score < 60) :
    grade = "E"
elif (score < 70) :
    grade = "D"
elif (score < 80) :
    grade = "C"
elif (score < 90) :
    grade = "B"
else :
    grade = "A"
```

Score	Grade
90-100	A
80-89	B
70-79	C
60-69	D
<60	E

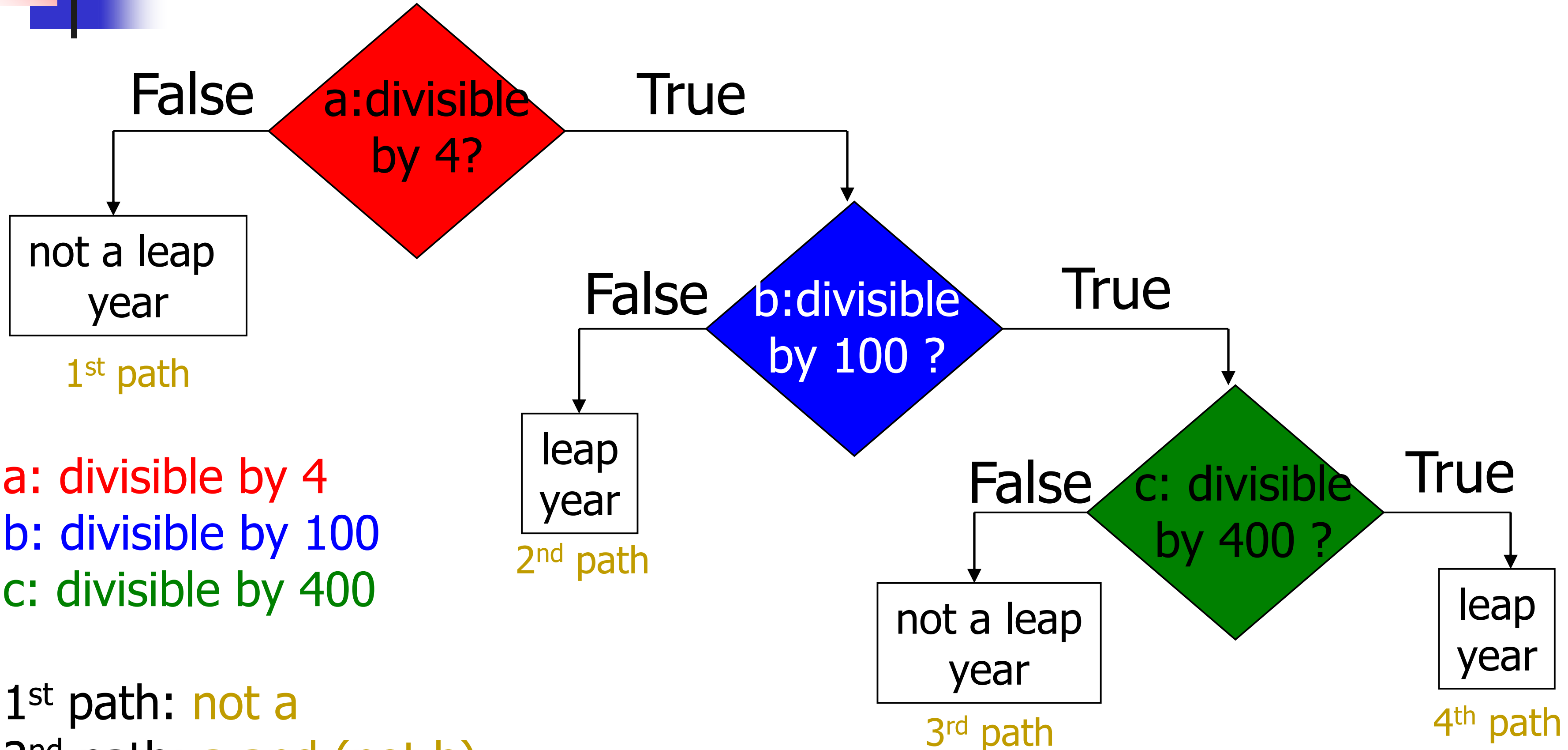
Alternative Solution to Quiz Grading

```
score = int(input("Enter the score: "))
grade = ""
if (score >= 90) :
    grade = "A"
else :
    if (score >= 80) :
        grade = "B"
    else :
        if (score >= 70) :
            grade = "C"
        else :
            if (score >= 60) :
                grade = "D"
            else :
                grade = "E"
```

Lab 7, Leap Year

- Obtain an integer from the keyboard. Print True if it specifies a leap year, otherwise print False.
- Usually years that are divisible by 4 are leap years.
  - Leap year: 1996, 2004, 2008, 2012, 2016
  - 1996 % 4 == 0, 2004 % 4 == 0, …
- However, years that are divisible by 100 are not leap years, unless the year is also divisible by 400.
  - Not leap year: 1900, as 1900 % 100 ==0, but 1900 % 400 !=0
  - Leap year: 2000, as 2000 % 400 == 0

Decision Tree



a: divisible by 4
b: divisible by 100
c: divisible by 400

1st path: not a
2nd path: a and (not b)
3rd path: a and b and (not c)
4th path: a and b and c

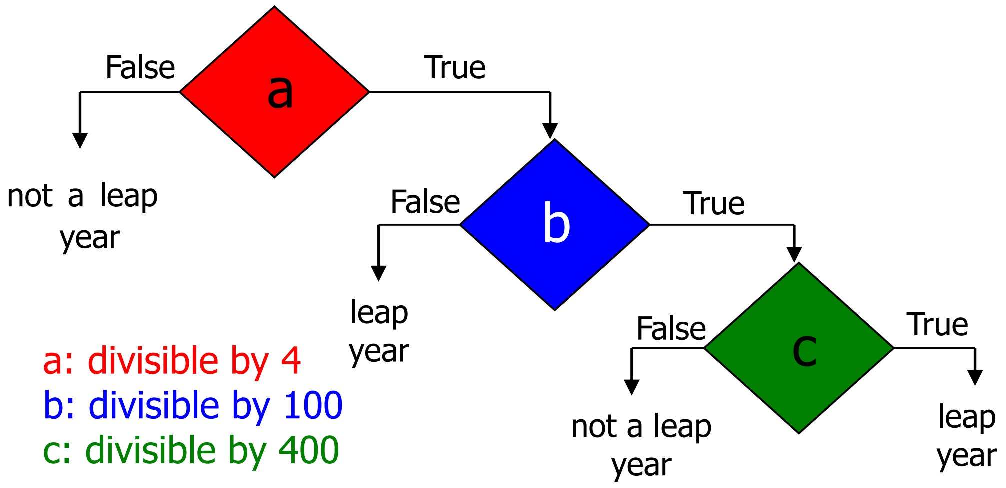
Solution to Leap Year

```
year = int(input("Enter the year:"))

a = (year%4 == 0)        # brackets are not essential
b = (year%100 == 0)
c = (year%400 == 0)
```

```
if (not a) :
      print("Not a leap year")
elif (not b) :
      print("Leap year")
elif (not c) :
      print("Not a leap year")
else:
      print("Leap year")
```

```
if (not a) :    # brackets are not essential
      print("Not a leap year")
else:
      if  (not b) :
                print("Leap year")
      else:
                if (not c)
                                print("Not a leap year")
                else:
                                print("Leap year")
```



a: divisible by 4
b: divisible by 100
c: divisible by 400

Boolean Test for a Leap Year

- It is a leap year if (a and (not b)) or (a and b and c)
- Equivalent solution:

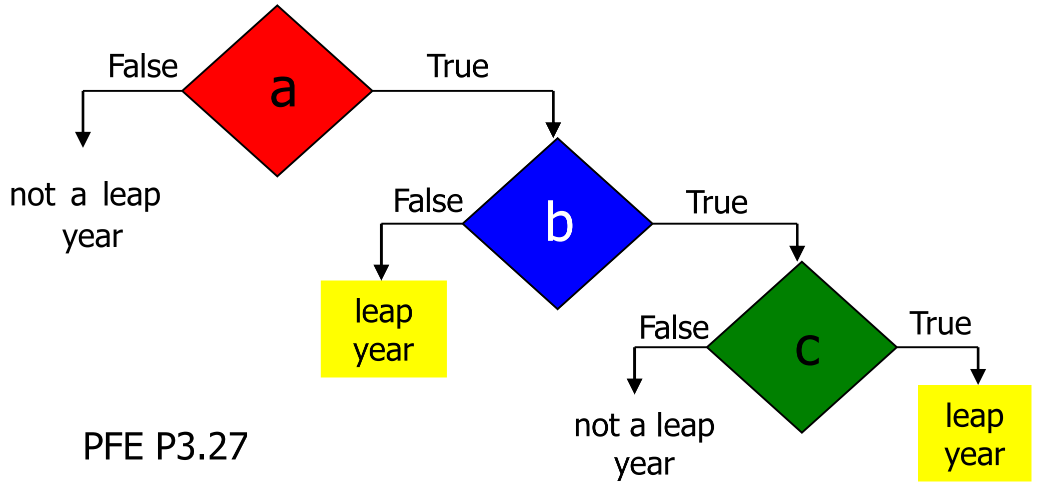
  a and ((not b) or (b and c))
- Proof of equivalence:

  case a = False (both are False)

  case a = True (both reduce to ((not b) or (b and c)))
- In this example only, (b and c) == c thus an equivalent solution is

  a and ((not b) or c)

a: divisible by 4
b: divisible by 100
c: divisible by 400



Solution to Leap Year

```
year = int(input("Enter the year:"))

a = (year%4 == 0)        # brackets are not essential
b = (year%100 == 0)
c = (year%400 == 0)

if a and ((not b) or c) :
    print("Leap year")
else :
    print("Not a leap year")
```

Solution to Leap Year

```
year = int(input("Enter the year:"))

a = (year%4 == 0)        # brackets are not essential
b = (year%100 == 0)
c = (year%400 == 0)

#a and ((not b) or c)

if (year%4==0) and (year%100 !=0 or year%400 == 0):
    print("Leap year")
else :
    print("Not a leap year")
```

Syntax for the while-Loop

```
while condition :
     statements
```

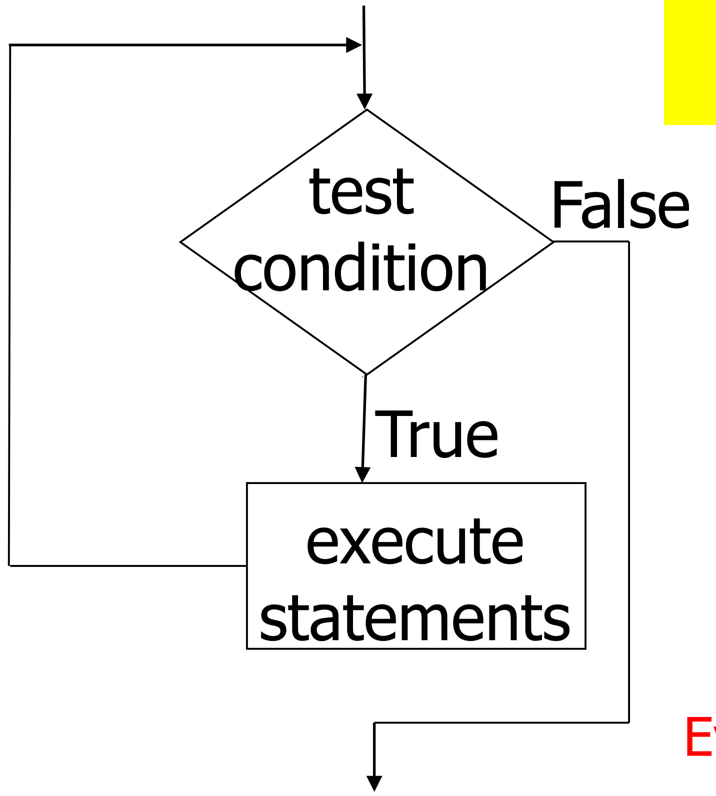
```
# If the value of the condition equals True,
# then the statements are executed
```

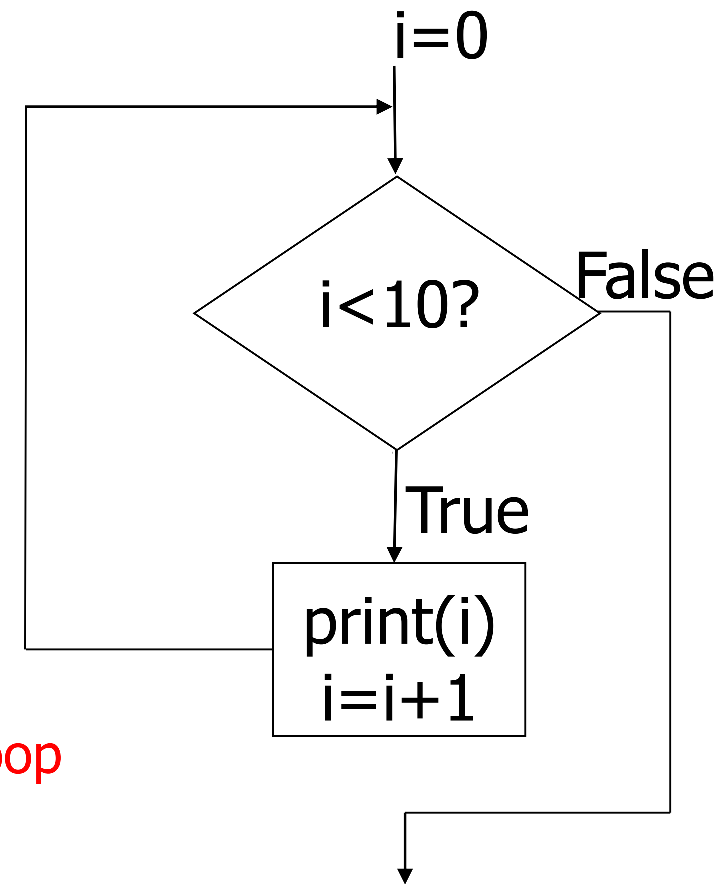
- Example:

  ```
  i = 0
  while i < 10 :
      print(i)
      i = i + 1
  ```

Flowchart for the while-Loop

```
i = 0
while i < 10 :
        print(i)
        i = i+1
```





Event controlled loop

Investment Problem Revisited

- You put £10,000 into a bank account that earns 5% interest per year.
- How many years does it take for the account balance to be double the original?
- (PFE, Section 1.7)
- Week 2

Example: compound interest

```
RATE = 5.0
INITIAL_BALANCE = 10000.0
TARGET = 2 * INITIAL_BALANCE
balance = INITIAL_BALANCE
year = 0

while (balance < TARGET) :
   year = year + 1
   interest = balance * RATE / 100
   balance = balance + interest

print("The investment doubled after", year, "years.")
```

Test Cases

- Use very simple test data to check that the while loop is correct.
- Eg. Set TARGET = INITIAL_BALANCE
- Eg. if

  RATE = 100.1%,

  TARGET = 2 * INITIAL_BALANCE

  then the balance is slightly more than doubled at the end of the first year.
- In both cases check the value of year on exiting the loop.

Example: compound interest

```
RATE = 5.0
INITIAL_BALANCE = 10000.0
TARGET = INITIAL_BALANCE
balance = INITIAL_BALANCE
year = 0

while (balance < TARGET) :
   year = year + 1
   interest = balance * RATE / 100
   balance = balance + interest

print("The investment reaches the target after",year,"years.")
```

What's the value of `year` on exiting the loop?

Example: compound interest

```
RATE = 100.1    #usury
INITIAL_BALANCE = 10000.0
TARGET = 2*INITIAL_BALANCE
balance = INITIAL_BALANCE
year = 0

while (balance < TARGET) :
   year = year + 1
   interest = balance * RATE / 100
   balance = balance + interest

print("The investment reaches the target after",year,"years.")
```

What's the value of `year` on exiting the loop?

while-Loop Examples

```
i = 0
total = 0
while i < 5 :
      total = total + i
      i = i + 1
      print(i, total)
```

```
i         total  # not printed
0         0      # not printed
1         0
2         1
3         3
4         6
5         10
```

```
i = 0
total = 0
while i < 5 :
      i = i + 1
      total = total + i
      print(i, total)
```

```
i         total  # not printed
0         0      # not printed
1         1
2         3
3         6
4         10
5         15
```

while-Loop Examples

```
i = 0
total = 0
while total < 10 :
     i = i + 1
     total = total + i
     print(i, total)
```

i	total	# not printed
0	0	# not printed
1	1	
2	3	
3	6	
4	10	

When `total` is 10, the loop condition is False and the loop ends.

while-Loop Examples

```
i = 0
total = 0
while total < 10 :
    i = i + 1
    total = total - i
    print(i, total)
```

```
i        total   # not printed
0        0        # not printed
1        -1
2        -3
3        -6
4        -10
…
```

Infinite loop

while-Loop Examples

```
i = 0
total = 0
while total < 0 :
    i = i+1
    total = total - i
    print(i, total)
```

No output

The statement `total < 0` is False when it is checked for the first time. The loop is never executed.

Infinite Loops

```
i = 0
total = 0
while total >= 0 :
   i = i+1
   total = total+i
print(i, total)
```

Wrong termination condition

```
year = 20
while year > 0:
    interest = balance * RATE / 100
    balance = balance + interest
```

Forget to change year

```
year = 20
while year > 0 :
    interest = balance * RATE / 100
    balance = balance + interest
    year = year + 1
```

year = year -1

The for-Loop For Strings

```
stateName = "Virginia"
for letter in stateName :
  print(letter)                #try print(letter, end="")
for ltr in stateName :
  print(ltr)    # the variable name can be changed (letter, ltr, etc)
# The successive values of letter are "V", "i", "r", etc.
# Output
# V
# i
# ...
```

range() function

`range([start], stop[, step])` It generates a sequence of integers

- `start:` Starting number of the sequence – 0 by default
- `stop:` Generate numbers up to, but not including this number
- `step:` Difference between each number in the sequence – 1 by default

```
range(1, 10, 2)
# 1, 3, 5, …, 9

range(1, 10)
# 1, 2, 3, …, 9

range(10)
# 0, 1, 2, …, 9
```

Count Controlled for-Loops

The loop iterates over a sequence of integers generated by `range()`

```
for i in range(1, 10) :             # i = 1, 2, 3, ..., 9
   print(i)


for i in range(1, 10, 2) :          # i = 1, 3, 5, ..., 9
    print(i)


for i in range(10) :                 # i = 0, 1, 2, ..., 9
   print(i)
```

Example of a for-Loop

```
RATE = 5.0
INITIAL_BALANCE = 10000.0

numYears = int(input("Enter number of years:"))
balance = INITIAL_BALANCE

for year in range(1, numYears+1) :
    interest = balance * RATE / 100
    balance = balance + interest
    print("%4d %10.2f" % (year, balance))
```

Output

```
Enter number of years: 10
   1  10500.00
   2  11025.00
   3  11576.25
   4  12155.06
   5  12762.82
   6  13400.96
   7  14071.00
   8  14774.55
   9  15513.28
  10  16288.95
```

for-Loop Examples

```
for i in range(10, 16):
```

# 10, 11, 12, …, 15 The ending value is never included in the sequence

```
for i in range(0, 11, 3):
```

# 0, 3, 6, 9   The third argument is the step value

```
for i in range(6):
```

# 0, 1, 2, 3, 4, 5   The loop is executed 6 times

```
for i in range(5, 0, -1) :
```

# 5, 4, 3, 2, 1   Use a negative step value to count down

```
for i in range(9,-3,-2):
```

# 9, 7 ,5, 3, 1, -1

Example of a for-Loop

- Read twelve temperature values (one for each month) and display the number of the month with the highest temperature

- Example: if the temperatures in degree C are

  18.2, 22.6, 26.4, 31.1, 36.6, 42.2, 45.7, 44.5, 40.2, 33.1, 24.2, 17.6

  then the program should display 7

- How to get the maximal number?

Example of a for-Loop

```
highestTemp = -273.15    # highest temperature, initially lowest temp
highestTempIndex = 0      # the month number of the highest temp

for i in range(1, 13):
    print('It's Month', i)
    temperature = float(input('Please input the temperature for this month:'))

    if highestTemp < temperature:
        highestTempIndex = i
        highestTemp = temperature

print('The hottest month is Month', highestTempIndex, 'with',
highestTemp,'degrees.')
```

Examples

- Write a loop that computes the sum of the squares of the numbers between 1 and 100, inclusive
- Use a single for loop to display a rectangle of asterisks with a given height and a given width
- Write a loop that computes the sum of all the odd digits in a non-negative integer n

Examples

- Write a loop that computes the sum of the squares of the numbers between 1 and 100, inclusive

```
squareSum = 0
for i in range(1,101):
    squareSum = squareSum + i * i
print('The sum of the squares of numbers in the range 1 to 100 is', squareSum)
```

Examples

- Use a single for-loop to display a rectangle of asterisks with a given height and a given width

```
width = int(input('Please input the width of a rectangle:'))
height = int(input('Please input the height of a rectangle:'))
if width < 0 or height < 0:
    print('width or height cannot be negative.')
else:
    for i in range(0, height):
        print('*' * width)
```

Examples

- Write a loop that computes the sum of all the odd digits in a non-negative integer n

```
nonNegIntStr = input('Please input a non-negative integer:')
nonNegInt = int(nonNegIntStr)
if nonNegInt < 0:
    print('The integer must be non-negative.')
else:
    oddSum = 0
    for digitStr in nonNegIntStr:
        digit = int(digitStr)
        if  digit%2 != 0:
            oddSum = oddSum + digit
    print('The sum of all the odd digits in', nonNegInt, 'is', oddSum)
```