Prerelease code

Stephen Yules 0061

Mr.Kieran

Lanna International School Thailand

Contents

[TASK 1 2](#_Toc70768578)

[Task 1.1 Pseudocode 2](#_Toc70768579)

[Task 1.1 Python 3](#_Toc70768580)

[Task 1.2 Pseudocode 4](#_Toc70768581)

[Task 1.2 Python 4](#_Toc70768582)

[Task 1.3 Pseudocode 5](#_Toc70768583)

[Task 1.3 Python 6](#_Toc70768584)

[Task 1.4 Pseudocode 7](#_Toc70768585)

[Task 1.4 Python 8](#_Toc70768586)

[Task 1.5.1 Pseudocode 9](#_Toc70768587)

[Task 1.5.1 Python 10](#_Toc70768588)

[Task 1.5.2 Pseudocode 11](#_Toc70768589)

[Task 1.5.2 Python 11](#_Toc70768590)

[Task 1.5.3 Pseudocode 12](#_Toc70768591)

[Task 1.5.3 Python 13](#_Toc70768592)

[Task 1.5.4 Pseudocode 14](#_Toc70768593)

[Task 1.5.4 Python 15](#_Toc70768594)

[Task 1.6 Pseudocode 16](#_Toc70768595)

[Task 1.6 Python 17](#_Toc70768596)

[Task 2 18](#_Toc70768597)

[Task 2.1 Pseudocode 18](#_Toc70768598)

[Task 2.1 Python 19](#_Toc70768599)

[Task 2.2 Pseudocode 20](#_Toc70768600)

[Task 2.2 Python 21](#_Toc70768601)

[Task 2.3 Pseudocode 22](#_Toc70768602)

[Task 2.3 Python 23](#_Toc70768603)

[Task 2.4 Pseudocode 24](#_Toc70768604)

[Task 2.4 Python 26](#_Toc70768605)

[Task 2.5 Pseudocode 28](#_Toc70768606)

[Task 2.5 Python 30](#_Toc70768607)

# TASK 1

## Task 1.1 Pseudocode

DECLARE students : ARRAY[1:9] OF STRING

DECLARE name, email, dob, spaces, student : STRING

DECLARE i, n, emaillen, namegap, emailgap : INTEGER

FOR i 🡨 1 TO LENGTH(students)

OUTPUT "Input the student's name."

INPUT name

OUTPUT "Input the student's email."

INPUT email

OUTPUT "Input the student's date of birth."

INPUT dob

students[i] 🡨 name & "\*" & email & "\*' & dob

ENDFOR

spaces 🡨 " "

OUTPUT “Here are all the students with headers.”

OUTPUT "Name", LEFT(spaces, 12), "Email", LEFT(spaces, 20), "Date of birth"

FOR i 🡨 1 TO LENGTH(students)

student 🡨 students[i]

n 🡨 1

WHILE MID(student, n, 1) <> "\*"

n 🡨 n + 1

ENDWHILE

name 🡨 LEFT(student, n - 1)

emaillen 🡨 LENGTH(students[i]) - LENGTH(name) - 11

email 🡨 MID(students[i], n + 1, emaillen)

dob 🡨 RIGHT(student, 9)

namegap 🡨 16 - LENGTH(name)

emailgap 🡨 25 - emaillen

OUTPUT name, LEFT(spaces, namegap), email, LEFT(spaces, emailgap), dob

ENDFOR

## Task 1.1 Python

# students : ARRAY[0:8]

# variables : ARRAY[0:2]

# name, email, dateOfBirth, spaces : STRING

# i, namegap, emailgap : INTEGER

students = []

for i in range(9):

students.append("")

for i in range(len(students)):

name = input("Input the student's name: ")

email = input("Input the student's email: ")

dateOfBirth = input("Input the student's date of the birth: ")

students[i] = name + "\*" + email + "\*" + dateOfBirth

spaces = "                                                      "

print("Name" + spaces[:12] + "Email" + spaces[:20] + "Date Of Birth")

for student in students:

variables = student.split("\*")

namegap = 16 - len(variables[0])

emailgap = 25 - len(variables[1])

print(variables[0] + spaces[:namegap] + variables[1] + spaces[:emailgap] + variables[2])

## Task 1.2 Pseudocode

PROCEDURE printstudents(students : ARRAY OF STRING)

DECLARE i : INTEGER

OUTPUT “Here are the students with their names, email and date of birth separated by \*.”

FOR i 🡨 1 TO LENGTH(students)

IF students[i] <> ""

THEN

OUTPUT students[i]

ENDIF

ENDFOR

ENDPROCEDURE

## Task 1.2 Python

# student : STRING

def printstudents(students):

print("Here are the students with their names, email and date of birth seperated by \*.")

for student in students:

if student != "":

print(student)

## Task 1.3 Pseudocode

DECLARE students : ARRAY[1:9] OF STRING

DECLARE stuname, email, dob, searchingname, tempname : STRING

DECLARE i, emaillen : INTEGER

DECLARE found : BOOLEAN

FOR i 🡨 1 TO LENGTH(students)

OUTPUT "Input the student's name."

INPUT stuname

OUTPUT "Input the student's email."

INPUT email

OUTPUT "Input the student's date of birth."

INPUT dob

students[i] 🡨 stuname & "\*" & email & "\*' & dob

ENDFOR

OUTPUT "Whose email would you like?”

INPUT searchingname

found 🡨 FALSE

i 🡨 1

WHILE found = FALSE AND i <= LENGTH(students)

tempname 🡨 LEFT(students[i], LENGTH(searchingname))

IF tempname = searchingname

THEN

emaillen 🡨 LENGTH(students[i]) - LENGTH(tempname) - 11

email 🡨 MID(students[i], n + 1, emaillen)

OUTPUT email

found 🡨 TRUE

ELSE

i 🡨 i + 1

ENDIF

ENDWHILE

IF found = FALSE

THEN

OUTPUT “No student found.”

ENDIF

## Task 1.3 Python

# students : ARRAY[0:8]

# variables : ARRAY[0:2]

# studentName, email, dateOfBirth, searchingname : STRING

# i : INTEGER

# found : BOOLEAN

students = []

for i in range(9):

students.append("")

for i in range(len(students)):

studentName = input("Input the student's name: ")

email = input("Input the student's email: ")

dateOfBirth = input("Input the student's date of the birth: ")

students[i] = studentName + "\*" + email + "\*" + dateOfBirth

searchingname = input("Input the name of the person whose email you want: ")

found = False

i = 0

while found == False and i < len(students):

variables = students[i].split("\*")

if variables[0] == searchingname:

print(variables[1])

found = True

else:

i = i + 1

if found == False:

print("No students found.")

## Task 1.4 Pseudocode

PROCEDURE searchbirthmonth(students : ARRAY OF STRING)

DECLARE matchinglist : ARRAY[1:9] OF STRING

DECLARE searchmonth, studate, printstring : STRING

DECLARE matchindex, index, n : INTEGER

DECLARE found : BOOLEAN

OUTPUT "Please state the first 3 letters of the month you want to search for, first letter capitalized."

INPUT searchmonth

matchindex 🡨 1

found 🡨 FALSE

FOR index 🡨 1 TO LENGTH(students)

studate 🡨 RIGHT(students[index], 7)

IF searchmonth = LEFT(studate, 3)

THEN

n 🡨 1

WHILE MID(students[index], n, 1) <> “\*”

n 🡨 n + 1

ENDWHILE

matchinglist[matchindex] 🡨 LEFT(students[index], n - 1)

matchindex 🡨 matchindex + 1

found 🡨 TRUE

ENDIF

ENDFOR

printstring 🡨 “”

IF found = FALSE

THEN

OUTPUT “No students found with this birth month.”

ELSE

FOR index 🡨 1 TO LENGTH(matchinglist)

IF matchinglist[index] <> “”

THEN

printstring 🡨 printstring & matchinglist[index] & “, ”

ENDIF

ENDFOR

OUTPUT LEFT(printstring, LENGTH(printstring) - 2)

ENDIF

ENDPROCEDURE

## Task 1.4 Python

# matchinglist : ARRAY[0:8]

# variables : ARRAY[0:2]

# searchmonth, student, studate : STRING

# found : BOOLEAN

def searchbirthmonth(students):

searchmonth = input("Input the first 3 letters of the month you want to search for, first letter  capitalized: ")

found = False

matchinglist = []

for student in students:

studate = student[-7:]

if searchmonth == studate[:3]:

variables = student.split("\*")

matchinglist.append(variables[0])

found = True

if found == False:

print("No students found.")

else:

print(matchinglist)

## Task 1.5.1 Pseudocode

DECLARE students : ARRAY[1:9, 1:5] OF STRING

DECLARE spaces : STRING

DECLARE i, namegap, emailgap, dobgap, studentIDgap : INTEGER

FOR i 🡨 1 TO LENGTH(students)

OUTPUT “Input the student’s name.”

INPUT students[i,1]

OUTPUT “Input the student’s email.”

INPUT students[i,2]

OUTPUT “Input the student’s date of birth.”

INPUT students[i,3]

OUTPUT “Input the student’s id.”

INPUT students[i,4]

OUTPUT “Input the student’s tutor id.”

INPUT students[i,5]

OUTPUT “Student completed.”

ENDFOR

OUTPUT “Here are all the students in a table with headers.”

spaces 🡨 “ “

OUTPUT "Name", LEFT(spaces, 12), "Email", LEFT(spaces, 20), "Date of birth", LEFT(spaces, 5), “StudentID”, LEFT(spaces, 5), “TutorID”

FOR i 🡨 1 TO LENGTH(students)

namegap 🡨 16 - LENGTH(students[i,1])

emailgap 🡨 25 - LENGTH(students[i,2])

dobgap 🡨 18 - LENGTH(students[i,3])

studentIDgap 🡨 14 - LENGTH(students[i,4])

OUTPUT students[i,1], LEFT(spaces, namegap), students[i,2], LEFT(spaces, emailgap), students[i,3], LEFT(spaces, dobgap), students[i,4], LEFT(spaces, studentIDgap), students[i,5]

ENDFOR

## Task 1.5.1 Python

# students : ARRAY[0:8,0:4]

# spaces : STRING

# i, namegap, emailgap, dobgap, studentIDgap : INTEGER

students = []

for i in range(9):

students.append([])

for i in range(9):

students[i].append(input("Input the student's name: "))

students[i].append(input("Input the student's email: "))

students[i].append(input("Input the student's date of birth: "))

students[i].append(input("Input the student's student ID: "))

students[i].append(input("Input the student's tutor ID: "))

print("")

print("Here are all the students in a table with headers.")

spaces = "                                                                "

print("Name" + spaces[:12] + "Email" + spaces[:20] + "Date of birth" + spaces[:5] + "StudentID" +  spaces[:5] + "TutorID")

for i in range(len(students)):

namegap = 16 - len(students[i][0])

emailgap = 25 - len(students[i][1])

dobgap = 18 - len(students[i][2])

studentIDgap = 14 - len(students[i][3])

print(students[i][0] + spaces[:namegap] + students[i][1] + spaces[:emailgap] + students[i][2]

             + spaces[:dobgap] + students[i][3] + spaces[:studentIDgap] + students[i][4])

## Task 1.5.2 Pseudocode

PROCEDURE printstudents(students : ARRAY OF STRING)

DECLARE rowindex : INTEGER

OUTPUT “Here are the students details.”

FOR rowindex 🡨 1 TO LENGTH(students)

IF students[rowindex, 1] <> “”

THEN

OUTPUT students[rowindex]

ENDIF

ENDFOR

ENDPROCEDURE

## Task 1.5.2 Python

# i : INTEGER

def printstudents(students):

print("Here are all the students.")

for i in range(len(students)):

if students[i][1] != "":

print(students[i])

## Task 1.5.3 Pseudocode

DECLARE students : ARRAY[1:9, 1:5] OF STRING

DECLARE searchingname : STRING

DECLARE i : INTEGER

DECLARE found : BOOLEAN

FOR i 🡨 1 TO LENGTH(students)

OUTPUT “Input the student’s name.”

INPUT students[i,1]

OUTPUT “Input the student’s email.”

INPUT students[i,2]

OUTPUT “Input the student’s date of birth.”

INPUT students[i,3]

OUTPUT “Input the student’s id.”

INPUT students[i,4]

OUTPUT “Input the student’s tutor id.”

INPUT students[i,5]

OUTPUT “Student completed.”

ENDFOR

OUTPUT “Who are you searching for?”

INPUT searchingname

found 🡨 FALSE

i 🡨 1

WHILE found = FALSE AND i <= LENGTH(students)

IF searchingname = students[i,1]

THEN

OUTPUT students[i,2]

found 🡨 TRUE

ELSE  
 i 🡨 i + 1

ENDIF

ENDWHILE

IF found = FALSE

THEN

OUTPUT “No student found.”

ENDIF

## Task 1.5.3 Python

# students : ARRAY[0:8,0:4]

# searchingname : STRING

# i : INTEGER

# found : BOOLEAN

students = []

for i in range(9):

students.append([])

for i in range(9):

students[i].append(input("Input the student's name: "))

students[i].append(input("Input the student's email: "))

students[i].append(input("Input the student's date of birth: "))

students[i].append(input("Input the student's student ID: "))

students[i].append(input("Input the student's tutor ID: "))

print("Student completed.")

searchingname = input("Who are you searching for: ")

found = False

i = 0

while found == False and i < len(students):

if students[i][0] == searchingname:

print(students[i][1])

found = True

else:

i = i + 1

if found == False:

print("No students found.")

## Task 1.5.4 Pseudocode

PROCEDURE searchbirthmonth(students : ARRAY OF STRING)

DECLARE matchinglist : ARRAY[1:9] OF STRING

DECLARE searchmonth, studate, printstring : STRING

DECLARE matchindex, i : INTEGER

DECLARE found : BOOLEAN

OUTPUT “Please state the first 3 letters of the month you want to search for, first letter capitalized.”

INPUT searchmonth

found 🡨 FALSE

matchindex 🡨 1

FOR i 🡨 1 TO LENGTH(students)

studate 🡨 students[i,3]

IF searchmonth = MID(studate, 4, 3)

THEN

matchinglist[matchindex] 🡨 students[i,1]

matchindex 🡨 matchindex + 1

found 🡨 TRUE

ENDIF

ENDFOR

printstring 🡨 “”

IF found = FALSE

THEN

OUTPUT “No students found with this birth month.”

ELSE

FOR i 🡨 1 TO LENGTH(matchinglist)

IF matchinglist[i] <> “”

THEN

printstring 🡨 printstring & matchinglist[i] & “, ”

ENDIF

ENDFOR

OUTPUT LEFT(printstring, LENGTH(printstring) - 2)

ENDIF

ENDPROCEDURE

## Task 1.5.4 Python

# matchinglist : ARRAY[0:8]

# searchmonth, studate : STRING

# i : INTEGER

# found : BOOLEAN

def searchbirthmonth(students):

searchmonth = input("Input the first 3 letters of the month you want to search for, first letter capitalized: ")

found = False

matchinglist = []

for i in range(len(students)):

studate = students[i][2]

studate = studate[-8:]

if searchmonth == studate[:3]:

matchinglist.append(students[i][0])

found = True

if found == False:

print("No students found with this birth month.")

else:

print(matchinglist)

## Task 1.6 Pseudocode

PROCEDURE search(students : ARRAY OF STRING)

DECLARE choice, searchstring : STRING  
DECLARE searchtype, i : INTEGER

DECLARE valid, found : BOOLEAN

OUTPUT “What type of data would you like to search for? Choose from name, email, date of birth, studentID or tutorID.”

valid 🡨 FALSE

searchtype 🡨 0

WHILE valid = FALSE

INPUT choice

CASE OF choice

“name” : searchtype 🡨 1

“email” : searchtype 🡨 2

“date of birth” : searchtype 🡨 3

“studentID” : searchtype 🡨 4

“tutorID” : searchtype 🡨 5

OTHERWISE : OUTPUT “Not a choice. Retry. ”

ENDCASE

IF searchtype <> 0

THEN

valid 🡨 TRUE

ENDIF

ENDWHILE

OUTPUT “Input the data you want to search for.”

INPUT searchstring

found 🡨 FALSE

FOR i 🡨 1 TO LENGTH(students)

IF searchstring = students[i, searchtype]

THEN

OUTPUT students[i]

found 🡨 TRUE

ENDIF

ENDFOR

IF found = FALSE

THEN

OUTPUT “No students found.”

ENDIF

ENDPROCEDURE

## Task 1.6 Python

# choice, searchstring : STRING

# searchtype, i : INTEGER

# valid, found : BOOLEAN

def search(students):

print("What type of data would you like to search for? Choose from name, email, date of birth,  studentID or tutorID.")

valid = False

searchtype = 9

while valid == False:

choice = input()

if choice == "name":

searchtype = 0

  elif choice == "email":

searchtype = 1

elif choice == "date of birth":

searchtype = 2

elif choice == "studentID":

searchtype = 3

elif choice == "tutorID":

searchtype = 4

else:

print("Not a choice. Retry inputting.")

if searchtype != 9:

valid = True

searchstring = input("Input the data you want to search for: ")

found = False

for i in range(len(students)):

if searchstring == students[i][searchtype]:

print(students[i])

found = True

if found == False:

print("No students found.")

# Task 2

## Task 2.1 Pseudocode

DECLARE studentID, email, dob : STRING

DECLARE continue : CHAR

continue 🡨 “y”

OPENFILE “students.txt” FOR WRITE

WHILE continue = “y”

OUTPUT “Input the student ID with 2 letters followed by 4 digits.”

INPUT studentID

OUTPUT “Input the student’s email.”

INPUT email

OUTPUT “Input the student’s date of birth in DDMMYY format.”

INPUT dob

WRITEFILE “students.txt”, studentID & email & dob & “\n”

OUTPUT “Would you like to continue? y/n”

INPUT continue

ENDWHILE

CLOSEFULE “students.txt”

## Task 2.1 Python

# stuID, email, dob : STRING

# keepgoing : CHAR

with open("students.txt","w") as f:

keepgoing = "y"

while keepgoing == "y":

stuID = input("Input the Student ID with 2 letters followed by 4 characters: ")

email = input("Input the email of the student: ")

dob = input("Input the date of birth with a DDMMYY format: ")

f.write(stuID + email + dob + "\n")

keepgoing = input("Would you like to continue? y/n: ")

## Task 2.2 Pseudocode

DECLARE searchID, fileline : STRING

DECLARE found : BOOLEAN

OUTPUT “Input the student ID you want to search for.”

INPUT searchID

found 🡨 FALSE

OPENFILE “students.txt” FOR READ

READLINE “students.txt”, fileline

WHILE NOT EOF(“students.txt”) AND found = FALSE

IF searchID = LEFT(fileline, 6)

THEN

OUTPUT MID(fileline, 7, LENGTH(fileline) - 12)

found 🡨 TRUE

ELSE

READLINE “students.txt”, fileline

ENDIF

ENDWHILE

IF found = FALSE

THEN

OUTPUT “No student found.”

ENDIF

CLOSEFILE “students.txt”

## Task 2.2 Python

# searchID, fileline, email : STRING

# found : BOOLEAN

searchID = input("Input the student ID: ")

found = False

with open("students.txt","r") as f:

fileline = f.readline()

while found == False and fileline != "":

if fileline[:6] == searchID:

email = fileline[6:(len(fileline)-7)]

print(email)

found = True

else:

fileline = f.readline()

if found == False:

print("No student found.")

## Task 2.3 Pseudocode

DECLARE searchID, fileline, stuID : STRING

DECLARE i : INTEGER

DECLARE found : BOOLEAN

OUTPUT “Input the substring you want to search for.”

INPUT searchID

found 🡨 FALSE

OPENFILE “students.txt” FOR READ

READLINE “students.txt”, fileline

WHILE NOT EOF “students.txt”

stuID 🡨 LEFT(fileline, 6)

FOR i 🡨 1 to (7 - LENGTH(searchID))

IF searchID = MID(stuID, i, LENGTH(searchID))

THEN

OUTPUT fileline

found 🡨 TRUE

ENDIF

ENDFOR

READLINE “students.txt”, fileline

ENDWHILE

IF found = FALSE

THEN

OUTPUT “No one was found.”

ENDIF

CLOSEFILE “students.txt”

## Task 2.3 Python

# searchID, fileline, stuID : STRING

# i : INTEGER

# found : BOOLEAN

found = False

searchID = input("Input substring: ")

with open("students.txt","r") as f:

fileline = f.readline()

while fileline != "":

stuID = fileline[:6]

for i in range(7-len(searchID)):

if searchID == stuID[i:i+len(searchID)]:

print(fileline)

found = True

fileline = f.readline()

if found == False:

print("No one found.")

## Task 2.4 Pseudocode

PROCEDURE addstudent()

DECLARE studentID, email, dob : STRING

DECLARE keepgoing : CHAR

keepgoing 🡨 “y”

OPENFILE “students.txt” FOR APPEND

WHILE keepgoing = “y”

OUTPUT “Input the student ID with 2 letters followed by 4 digits.”

INPUT studentID

OUTPUT “Input the student’s email.”

INPUT email

OUTPUT “Input the student’s date of birth in DDMMYY format.”

INPUT dob

WRITEFILE “students.txt”, studentID & email & dob & “\n”

OUTPUT “Would you like to continue? y/n”

INPUT keepgoing

ENDWHILE

CLOSEFILE “students.txt”

PROCEDURE searchstudent()

DECLARE searchID, fileline : STRING

DECLARE found : BOOLEAN

OUTPUT “Input the substring you want to search for.”

INPUT searchID

found 🡨 FALSE

OPENFILE “students.txt” FOR READ

READLINE “students.txt”, fileline

WHILE NOT EOF “students.txt”

IF searchID = LEFT(fileline, LENGTH(searchID))

THEN

OUTPUT fileline

found 🡨 TRUE

ENDIF

READLINE “students.txt”, fileline

ENDWHILE

IF found = FALSE

THEN

OUTPUT “No student found.”

ENDIF

CLOSEFILE “students.txt”

DECLARE command : STRING

DECLARE continue : CHAR

continue 🡨 “y”

WHILE continue = “y”

OTUPUT “Enter add or search to either add a student or search a studentID.”

INPUT command

CASE OF command

“add” : CALL addstudent()

“search” : CALL searchstudent()

OTHERWISE : OUTPUT “Not a valid command.”

ENDCASE

OUTPUT “Would you to perform another action? y/n”

INPUT continue

ENDWHILE

## Task 2.4 Python

def addstudent():

# stuID, email, dob : STRING

# nextstudent : CHAR

with open("students.txt","a") as f:

nextstudent = "y"

while nextstudent == "y":

stuID = input("Input the Student ID with 2 letters followed by 4 characters: ")

email = input("Input the email of the student: ")

dob = input("Input the date of birth with a DDMMYY format: ")

f.write(stuID + email + dob + "\n")

nextstudent = input("Would you like to continue? y/n: ")

def searchstudent():

# searchID, fileline, stuID : STRING

# i : INTEGER

# found : BOOLEAN

found = False

searchID = input("Input substring: ")

with open("students.txt","r") as f:

fileline = f.readline()

while fileline != "":

stuID = fileline[:6]

for i in range(7-len(searchID)):

if searchID == stuID[i:i+len(searchID)]:

print(fileline)

found = True

fileline = f.readline()

if found == False:

print("No one found.")

# command : STRING

# keepgoing : CHAR

keepgoing = "y"

while keepgoing == "y":

command = input("Enter add or search to either add a student or search for a studentID: ")

if command == "add":

addstudent()

elif command == "search":

searchstudent()

else:

print("Command not available.")

keepgoing = input("Would you like to input another command? y/n: ")

## Task 2.5 Pseudocode

DECLARE stuID, email, dob : STRING

DECLARE validcount, i : INTEGER

DECLARE valid : BOOLEAN

valid 🡨 FALSE

WHILE valid = FALSE

OUTPUT “Input the student ID, 2 letters followed by 4 digits.”

INPUT stuID

IF LENGTH(stuID) <> 6

THEN

OUTPUT “Not right length. Retry.”

ELSE

validcount 🡨 0

FOR i 🡨 1 TO 2

IF MID(stuID, i, 1) >= “A” AND MID(stuID, i, 1) <= “Z”

THEN

validcount 🡨 validcount + 1

ENDIF

ENDFOR

FOR i 🡨 1 TO 4

IF MID(stuID, 2 + i, 1) >= “0” AND MID(stuID, 2 + i, 1) <= “9”

THEN

validcount 🡨 validcount + 1

ENDIF

ENDFOR

IF validcount = 6

THEN

valid 🡨 TRUE

ELSE

OUTPUT “Invalid student ID. Retry.”

ENDIF

ENDIF

ENDWHILE

Output “Input the student’s email.”

INPUT email

valid 🡨 FALSE

WHILE valid = FALSE

OUTPUT “Input the date of birth in DDMMYY format.”

INPUT dob

IF LENGTH(dob) <> 6

THEN

OUTPUT “Incorrect length. Retry.”

ELSE

validcount 🡨 0

IF MID(dob, 1, 1) >= “0” AND MID(dob, 1, 1) <= “3”

THEN

validcount 🡨 validcount + 1

ENDIF

IF MID(dob, 2, 1) >= “0” AND MID(dob, 2, 1) <= “9”

THEN

validcount 🡨 validcount + 1

ENDIF

IF MID(dob, 3, 1) >= “0” AND MID(dob, 3, 1) <= “1”

THEN

validcount 🡨 validcount + 1

ENDIF

FOR i 🡨 1 TO 3

IF MID(dob, i + 3, 1) >= “0” AND MID(dob, i + 3, 1) <= “9”

THEN

validcount 🡨 validcount + 1

ENDIF

ENDFOR

IF validcount = 6

THEN

valid 🡨 TRUE

ELSE

OUTPUT “Invalid date. Retry.”

ENDIF

ENDIF

ENDWHILE

OPENFILE “students.txt” FOR APPEND

WRITEFILE “students.txt”, stuID & email & dob & “\n”

CLOSEFILE “students.txt”

## Task 2.5 Python

# stuID, email, dob : STRING

# validcount, i : INTEGER

# valid : BOOLEAN

valid = False

while valid == False:

stuid = input("Input the Student ID with 2 letters followed by 4 characters: ")

if len(stuID) != 6:

print("Not right length.")

else:

validcount = 0

for i in range(2):

if stuID[i] >= "A" and stuID[i] <= "Z":

validcount = validcount + 1

for i in range(2,6):

if stuID[i] >= "0" and stuID[i] <="9":

validcount = validcount + 1

if validcount == 6:

valid = True

else:

print("Invalid ID. Retry.")

email = input("Input the email of the student: ")

valid = False

while valid == False:

dob = input("Input the date of birth with a DDMMYY format: ")

if len(dob) != 6:

print("Not right length. Retry.")

else:

if dob[0] < "0" or dob[0] > "3":

print("Invalid date of birth. Retry.")

elif dob[1] < "0" or dob[1] > "9":

print("Invalid date of birth. Retry.")

elif dob[2] < "0" or dob[2] > "1":

print("Invalid date of birth. Retry.")

elif dob[3] < "0" or dob[3] > "9":

print("Invalid date of birth. Retry.")

elif dob[4] < "0" or dob[4] > "9":

print("Invalid date of birth. Retry.")

elif dob[5] < "0" or dob[5] > "9":

print("Invalid date of birth. Retry.")

else:

valid = True

with open("students.txt", "a") as f:

f.write(stuID + email + dob + "\n")