Cover sheet

Student name: Vu Nam Phuong
Student ID: 54781288

```
#2018-08-17  Nam Vu  <npv14@cs16079ho>
from socket import *
from datetime import *
import sys
import select

def checkPacket(data):
    """This function is using for check the packet request as the require"""
    valid = True
    if len(data) != 6:
        valid = False
    if data[0:2].hex() != '497e':
        valid = False
    if data[2:4].hex() != '0001':
        valid = False
    if data[4:6].hex() != '0001' and data[4:6].hex() != '0002' :
        valid = False
    return valid

def dtResponse(dateOrTime, language, year, month, day, hour, minute):
    """This function is using for pack the packet response as the require"""
    pack = bytearray();
    pack += int(0x497e).to_bytes(2, byteorder='big')
    pack += int(0x0002).to_bytes(2, byteorder='big')
    if language == 1:
        pack += int(0x0001).to_bytes(2, byteorder='big')
    elif language == 2:
        pack += int(0x0002).to_bytes(2, byteorder='big')
    else:
        pack += int(0x0003).to_bytes(2, byteorder='big')
    pack += int(year).to_bytes(2, byteorder='big')
    pack += int(month).to_bytes(1, byteorder='big')
    pack += int(day).to_bytes(1, byteorder='big')
    pack += int(hour).to_bytes(1, byteorder='big')
    pack += int(minute).to_bytes(1, byteorder='big')
    #create dictionary to store the month's name of English, Maori and German
    dictMonthE = {1:"January", 2:"February", 3:"March", 4:"April", 5:"May" ,6:"June"
, 7:"July", 8:"August", 9:"September", 10:"October", 11:"November" ,12:"December" }
    dictMonthM = {1:"Kohit¯atea", 2:"Hui-tanguru", 3:"Pout¯u-te-rangi", 4:"Paenga-wh
¯awh¯", 5:"Haratua" ,6:"Pipiri", 7:"H¯ongongoi", 8:"Here-turi-k¯ok¯a", 9:"Mahuru", 1
0:"Whiringa-¯a-nuku", 11:"Whiringa-¯a-rangi" ,12:"Hakihea" }
    dictMonthG = {1:"Januar", 2:"Februar", 3:"M¨arz", 4:"April", 5:"Mai" ,6:"Juni",
7:"Juli", 8:"August", 9:"September", 10:"Oktober", 11:"November" ,12:"Dezember" }
    if language == 1:
        monthText = dictMonthE[month];
    elif language == 2:
        monthText = dictMonthM[month];
    elif language == 3:
        monthText = dictMonthG[month];
    #create the return text field with an appropriate language
    if dateOrTime == '0001':
        if language == 1:
            text = "Today¿s date is {} {}, {}".format(monthText, day, year)
        elif language == 2:
            text = "Ko te ra o tenei ra ko {} {}, {}".format(monthText, day, year)
        else:
            text = "Heute ist der {}. {} {}".format(day, monthText, year)
    else:
        if language == 1:
            text = "The current time is {}:{}".format(hour, minute)
        elif language == 2:
            text = "Ko te wa o tenei wa {}:{}".format(hour, minute)
        else:
            text = "Die Uhrzeit ist {}:{}".format(hour, minute)
    #Encode the text in to bytes Using String.encode('utf8')
    encodeText = text.encode('utf8')
    #Check and count the length of the Text field
    length = len(encodeText)
    if length <=  225:
        pack += int(length).to_bytes(1, byteorder='big')
        pack += encodeText
        print("Successul pack Data Response")
    else:
        print("The text exceeds 255 bytes")
    return pack

def main(englishPortNo, maoriPortNo, germanPortNo):
    #create Server UDP socket and bind with appropriate Port number
    try:
        serverSocketE = socket(AF_INET, SOCK_DGRAM)
        serverSocketE.bind(("127.0.1.1", englishPortNo))
        serverSocketM = socket(AF_INET, SOCK_DGRAM)
        serverSocketM.bind(("127.0.1.1", maoriPortNo))
        serverSocketG = socket(AF_INET, SOCK_DGRAM)
        serverSocketG.bind(("127.0.1.1", germanPortNo))
    except:
        print("There is something wrong with the UDP socket")
    while True:
        #Waiting for the packet request
        reader,_, _ = select.select([serverSocketE, serverSocketM, serverSocketG], [
], [])
        for i in reader:
            if i is serverSocketE:
                #found another socket connect to English Socket
                data = i.recvfrom(1024)
                ipSender = data[1]
                print("Recived request from server Socket English at IP:", ipSender)
                packetByteArray = bytearray(data[0])
                #Check the request packet and prepare the Packet response and send b
ack to the client
                if(checkPacket(packetByteArray)):
                    timeCurrent = str(datetime.now().time()).split(':')
                    dateToday = str(datetime.now().date()).split('-')
                    packet = dtResponse(packetByteArray[4:6].hex(), 1, dateToday[0],
 int(dateToday[1]), dateToday[2], timeCurrent[0], timeCurrent[1])
                    print(packet)
                    serverSocketE.sendto(packet,ipSender)
                    print("Sent Response Packet")
                else:
                    print("Invalid Packet")
                print("-----------------------------------------------------------------
-------------------")
            if i is serverSocketM:
                #found another socket connect to Maori Socket
                data = i.recvfrom(1024)
                ipSender = data[1]
                print("Recived request from server Socket Maori at IP:", ipSender)
                packetByteArray = bytearray(data[0])
                #Check the request packet and prepare the Packet response and send b
ack to the client
                if(checkPacket(packetByteArray)):
                    timeCurrent = str(datetime.now().time()).split(':')
                    dateToday = str(datetime.now().date()).split('-')
                    packet = dtResponse(packetByteArray[4:6].hex(), 2, dateToday[0],
 int(dateToday[1]), dateToday[2], timeCurrent[0], timeCurrent[1])
                    print(packet)
                    serverSocketE.sendto(packet,ipSender)
                    print("Sent Response Packet")
                else:
                    print("Invalid Packet")
                print("-----------------------------------------------------------------
-------------------")
            if i is serverSocketG:
                #found another socket connect to German Socket
                data = i.recvfrom(1024)
                ipSender = data[1]
                print("Recived request from server Socket German at IP:", ipSender)
                packetByteArray = bytearray(data[0])
                #Check the request packet and prepare the Packet response and send b
ack to the client
                if(checkPacket(packetByteArray)):
                    timeCurrent = str(datetime.now().time()).split(':')
                    dateToday = str(datetime.now().date()).split('-')
                    packet = dtResponse(packetByteArray[4:6].hex(), 3, dateToday[0],
 int(dateToday[1]), dateToday[2], timeCurrent[0], timeCurrent[1])
                    print(packet)
                    serverSocketE.sendto(packet,ipSender)
                    print("Sent Response Packet")
                else:
                    print("Invalid Packet")
                print("-----------------------------------------------------------------
--------------------")

def start():
    """This function is using for start the program"""
    englishPortNo = int(sys.argv[1])
    maoriPortNo = int(sys.argv[2])
    germanPortNo = int(sys.argv[3])
    if englishPortNo != maoriPortNo and maoriPortNo != germanPortNo and englishPortN
o != germanPortNo and 1024 < englishPortNo < 64000 and 1024 < maoriPortNo < 64000 an
d 1024 < germanPortNo < 64000:
        main(englishPortNo, maoriPortNo, germanPortNo)
    else:
        print("Input Error the Server is closed")
start()

"""
Command codes for testing the program:
    python3 Server.py 63999 63998 63997
"""
```

```
#2018-08-17  Nam Vu  <npv14@cs16079ho>
from socket import *
import select
import sys
def dtRequest(dateOrTime):
    """This function is using for create the Request Packet that pack the number req
uire into byte array"""
    pack = bytearray();
    pack += int(0x497e).to_bytes(2, byteorder='big')
    pack += int(0x0001).to_bytes(2, byteorder='big')
    if dateOrTime == "date":
        pack += int(0x0001).to_bytes(2, byteorder='big')
    if dateOrTime == "time":
        pack += int(0x0002).to_bytes(2, byteorder='big')
    print("Successful pack data request")
    return pack
def checkRecivedPacket(data):
    """This function is using for check if the Packet recived is Valid or not as the
 require"""
    valid = 0
    if len(data) >= 13:
        valid += 1
    if data[0:2].hex() == '497e':
        valid += 1
    if data[2:4].hex() == '0002':
        valid += 1
    if data[4:6].hex() == '0001' or data[4:6].hex() == '0002' or data[4:6].hex() ==
'0003' :
        valid += 1
    if data[6:8].hex() != '0002':
        valid += 1
    if int.from_bytes(data[6:8], 'big') < 2100:
        valid += 1
    if 1 <= int.from_bytes(data[8:9], 'big') <= 12:
        valid += 1
    if 1 <= int.from_bytes(data[9:10], 'big') <= 31:
        valid += 1
    if 0 <= int.from_bytes(data[10:11], 'big') <= 23:
        valid += 1
    if 0 <= int.from_bytes(data[11:12], 'big') <= 59:
        valid += 1
    if len(data) == 13 + int.from_bytes(data[12:13], 'big'):
        valid += 1
    if valid == 11:
        print("Packet is valid")
    else:
        print("Packet is invalid")
    return valid == 11


def main(dateOrTime, ipORDomain, portNo):
    #creat Client UDP socket
    print("Start main")
    clientSocket = socket(AF_INET, SOCK_DGRAM)
    #Packed the data request
    packRequest = dtRequest(dateOrTime)
    try:
        clientSocket.sendto(packRequest, (ipORDomain, portNo))
        print("Sent request packet")
    except:
        print("Did not find the Server")
    found = False
    #Send the request packet and waiting for response in 1 second
    reader,_, _ = select.select([clientSocket], [], [], 1)
    for i in reader:
            if i is clientSocket:
                found = True
                data = i.recvfrom(1024)
                print("Recived data response")
                packetRecived  = bytearray(data[0])
                if checkRecivedPacket(packetRecived):
                    #Print the Text field of the packet response. It will look like
:
```

```
                                """
                                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                >>>>>>>>>> Today¿s date is August 17, 2018 <<<<<<<<<<<
                                ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                                """
                                print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~")
                                print('>>>>>>>>>>',packetRecived[13:].decode('utf8'), '<<<<<<<<
<<<')
                                print("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~")
    clientSocket.close()
    if not found:

        print("Packet loss")
        print("Please check IP and port number of Server")
    print("Program's closed")




def start():
    """This function is using for start the program if all arguments are valid then
start the main funtion"""
    valid = True
    try:
        dateOrTime = sys.argv[1]
        if dateOrTime != "date" and dateOrTime != "time":
            valid = False
            print("Invalid input")
    except:
        print("Invalid input")

    if valid:
        try:
            ipORDomain = sys.argv[2]
            check = ipORDomain.split(".")
            if len(check) == 4:
                for i in check:
                    if int(i) >= 256 or int(i) < 0:
                        valid = False
                        break;
                if not valid:
                    print("Invalid IP")
            else:
                try:
                    #Conver the hostname to IP then check the hostname is valid or n
ot
                    ip = getaddrinfo(ipORDomain, 'www')
                    print("ip: ", ip)
                    ipORDomain = ip[0][4][0]
                except:
                    print("Invalid Hostname")
                    valid = False
        except:
            print("Invalid input")
    if valid:
        try:
            portNo= int(sys.argv[3])
            if not (1024 < portNo < 64000):
                print("Invalid port number")
                valid = False
        except:
            print("Invalid input")


    if valid:
        main(dateOrTime, ipORDomain, portNo)

start()
"""
Command codes for testing the program:
    // Date:
```

```
    python3 Client.py date 127.0.1.1 63999
    python3 Client.py date 127.0.1.1 63998
    python3 Client.py date 127.0.1.1 63997
    // Time:
    python3 Client.py time 127.0.1.1 63999
    python3 Client.py time 127.0.1.1 63998
    python3 Client.py time 127.0.1.1 63997
"""
```

Plagiarism Declaration

This form needs to accompany your COSC 264 assignment submission.

I understand that plagiarism means taking someone else's work (text, program code, ideas, concepts) and presenting them as my own, without proper attribution. Taking someone else's work can include verbatim copying of text, figures/images, or program code, or it can refer to the extensive use of someone else's original ideas, algorithms or concepts.

I hereby declare that:

- My assignment is my own original work. I have not reproduced or modified code, figures/images, or writings of others without proper attribution. I have not used original ideas and concepts of others and presented them as my own.
- I have not allowed others to copy or modify my own code, figures/images, or writings. I have not allowed others to use original ideas and concepts of mine and present them as their own.
- I accept that plagiarism can lead to consequences, which can include partial or total loss of marks, no grade being awarded and other serious consequences, including notification of the University Proctor.

Name:	VU NAM PHUONG
Student ID:	54781288
Signature:	[illegible]
Date:	17/08/2018