CSE-A1111 Basic Course in Programming Y1 Exam 28.5.2015

Write the following information clearly on top of each paper you submit: name of the course, date of the exam, your full name, student ID, the total number of papers you submit, and your signature.

Important instructions: Use indentations of the length of two squares in your code. If your indentations are not clear enough, you lose points. You do not have to write any comments in your code. You can assume that the input given by the user is correct, if it is not told in the problem that you should handle the incorrect input. **Calculators and any extra material are not allowed. However, students whose mother tongue is not Finnish may use a dictionary, if it does not contain any extra markings. Those students may also obtain both Finnish and English / Swedish exam sheet, if they want.**

1. Parts a, b, c and d: What is printed when the given Python program is executed? It is enough to give the correct output without any explanations. Parts e, f, and g: Explain, in plain English using a couple of sentences, the *purpose* of the given Python function. (Do not say *how* the code works. Instead say what the function would be used for, for example "the function calculates and returns the sum of the integers in the list given as a parameter".) Notice that some of the programs or functions may contain errors. In that case, explain what the incorrect program prints or how the incorrect function works.

a) (2 p)

```
def main():
    temperature = 28
    if temperature > 10:
        print("Warm.")
    elif temperature >= 25:
        print("Hot.")
    else:
        print("Cold.")

main()
```

b) (2 p)

```
def main():
    price = 100
    reduction = 10
    new_price = price - reduction
    reduction = 30
    print(new_price)

main()
```

c) (3 p)

```
def main():
    list1 = [10, 20, 30, 60, 40]
    result = 100
    i = 0
    while i < len(list1):
        if result > 50:
            result = result - list1[i]
        i = i + 1
    print(result)

main()
```

d) (5 p)

```
def change(number, numbers):
    numbers[1] = 115
    number = 46
    return number

def main():
    counter = 33
    list1 = [10, 20, 30]
    change(counter, list1)
    print(counter)
    for element in list1:
        print(element)

main()
```

e) You may assume that the parametres of the function are two equally-long lists of integers. (4 p)

```
def mystery1(list1, list2):
    result = []
    i = 0
    while i < len(list1):
        if list1[i] < list2[i]:
            result.append(list1[i])
        else:
            result.append(list2[i])
        i += 1
    return result
```

f) You may assume that the first parameter of the function is a string and the second parameter is a positive integer. (4 p)

```
def mystery2(str1, number):
    if number > len(str1):
        return str1 + "*" * (number - len(str1))
    else:
        return str1[0:number]
```

g) You may assume that the first parameter is a list containing positive integers and the second parameter is a positive integer. (5 p)

```
def mystery3(list1, number):
    i = 0
    while i < len(list1):
        if (list1[i] % number) != 0:
            return False
        i += 1
    return True
```

2. a) You need mobile broadband for your laptop computer. You also need a USB wireless modem (Internet stick). The operator gives 5 per cent reduction from the normal monthly price if you make a contract for at least 12 months. In addition, the operator gives you a free USB modem if you make a contract for at least 24 months. Otherwise, you must buy the USB modem yourself. Write a Python program which helps you to estimate the costs of the different contract periods. The program should ask the user to input the normal monthly price (without reductions and USB modem), the price of the USB modem and the contract period (in months). The program should calculate and output monthly costs when both the monthly operator price (with possible reductions) and the price of the USB modem (the price of the modem divided by the length of the contract period) are taken into account. If the operator gives a free modem, the price of the modem is not taken into account. (10 p.)

   b) A firm pays to its sellers 50 euros for each day. In addition, if the total sale of the seller on a certain day exceeds a limit set by the firm, the seller receives an extra bonus for that day. The amount of the extra bonus is 15 per cent of the seller's total sale on that day. Assume that the daily sales of a certain seller has been stored in a list such that the 1st element is the total sale of the 1st day, the 2nd element is the total sale of the 2nd date and so on. Write a Python function `calculate_pay(daily_sales, limit)`, which calculates and returns the total sum the firm should pay to a seller. The value returned is one decimal number (the sum of the daily pays when both basic pay and extra bonuses are taken into account). Write only this function, not any other parts of the program. (20 p)

3. A firm has a text file containing information about its products, the normal prices of the products and the discounts (per cent) of the products. Each line of the file contain information about one product, and the different fields (name of the product, the normal price, and the reduction) are separated by a semicolon. The lines of the file could be as follows:

```
Pan Hackman;40.0;15.0
Brush;8.0;5.0
New Cola;2.0;20.0
```

   Write a Python program which asks the user to input the name of the file. The program reads that file and ouputs the list of the names and the reduced prices of those products whose reductions is at least 10 per cent. If the reduction of a product is less than 10 per cent, no information is printed on that product. In the example above, the program should output

```
Pan Hackman 34.0
New Cola 1.6
```

   The program must be able to handle the following errors

   - The file does not exist or it is not possible to read the file because of some other reason.
   - In some line, the price or reduction of the product cannot be converted to a decimal number.

   In these cases, the program must output the type of the error occured and stop. The program does not have to continue reading after the defective line. You may also assume that each line of the file consists of three fields separated by a semicolon. Your program does not have to handle cases where the file contains empty lines or lines consisting of only one or two fields, for example. (20 p)

The last problem is on the next page

4. A firm has a game in its web page. The user can play the game alone (against the computer) and he/she is given points according to the result (the higher points the better). The firm wants to maintain statistics of the games played. However, if the player wants that his/her results are included in the statistics, he/she has to buy a licence for the game. Write a class `Player` to describe one player .

   `Player` object must have the following data attributes (fields):
   - `__name` name of the player
   - `__licence` the value of this attribute is `True`, if the player has a valid licende, and `False` otherwise.
   - `__number_of_games` the number of the games this player has played
   - `__record` the record of this player (the highest points he/she has achieved in his/her games)
   - `__total_points` the total number of points the player has achieved in all games he/she has played.

   Write the following methods in your class. (To shorten your solution, some methods which would be useful have been omitted. If the description does not say anything about the return value of a method, the method does not have to return anything.)
   - `__init__(self, player_name)` creates a new `Player` object. The name of the player is given as the parameter. The record, number of games and total points of the new player are 0. The licence of the new player is valid.
   - `get_record(self)` returns the record of this player.
   - `has_licence(self)` returns `True`, if the player has a valid licence, and `False` otherwise.
   - `add_licence(self)` changes the licence of the player to be valid.
   - `remove_licence(self)` changes the licence of the player to be invalid.
   - `add_game(self, points)` adds information of the new game result to this player, if she/he has a valid licence. The points of the game are given as a parameter. The method updates the number of games and the total points and also the record if necessary. If the licence is not valid, the method does no updates.
   - `calculate_average(self)` calculates and returns the average number of points this player has obtained from all his/her games. If the player has played no games yet, the method returns 0.0.
   - `master_player(self)` returns value `True` if this player is a master player and `False` otherwise. The player is a master player if his/her record is at least 4500 points and he/she has played at least 30 games (the both conditions must be true). Otherwise, he/she is not a master player.
   - `__str__(self)` returns a string which contains the name, record and number of games of this player and either the text "licence valid" or "licence not valid" according to the information whether the licence is valid or not .

   In addition, write a main function which creates two `Player` objects, calls two times `add_game` method for both players and after that once `get_record` method for the second player. The program must also output the record returned by the function. Next, the program must find out if the second player is a master player and output either "The player is a master player" or "The player is not a master player" accordingly. Then the program must remove the licence of the second player. Finally, the program has to output basic information (the name, the record, the number of games and whether the licence is valid) about both players. You may choose the names of the players and the results of the games yourself. Your program does not have to ask any input from the user. Your main program may either be in the same module with your class or in a separate module (25 p)