HÖGSKOLAN VÄST

TENTAMEN GPP100/GPP101

Grundläggande programmering med Python (och IoT)

Datum	25 Augusti 2023
Tid	8:30 – 12:30
Examinator	Darwen Al-Taeshi
Lärare Besök	Darwen Al-Taeshi
Hjälpmedel	Ja
Antal uppgifter	kursboken 6
Antal sidor Max	4
poäng	30
Betygsgränser	3: 15 – 19 poäng 4: 20 – 24 poäng 5: 25 – 30 poäng
Övrigt Resultat	Glöm inte att testa din kod. Skriv egna test för att kom- plettera de givna testfallen.
anslås	senast 15 September 2023

Skriv din anonymitetskod i varje inlämnad fil. Lämna in filerna utan att ändra deras namn. Lägg dem i en katalog som har din anonymitetskod som namn.

1. (5p)

I ett virtuellt husdjurssimuleringsspel tar spelare hand om husdjur som kallas "BitPets." BitPets måste matas regelbundet för att hålla dem friska och glada. Matningsplanen är som följer: BitPets måste matas var 4:e timme. Men om en BitPet fortfarande är hungrig efter 5 timmar sedan dess senaste måltid blir den olycklig.

Skriv en Python-funktion som heter feeding_status som tar en heltalsvariabel hours_since_last_meal som inmatning och returnerar en sträng som indikerar BitPets matningsstatus. Funktionen bör returnera:

"Full" om BitPet har matats inom de senaste 4 timmarna. "Hungrig" om BitPet har matats mellan 4 och 5 timmar sedan. "Olycklig" om BitPet har matats för mer än 5 timmar sedan.

```
def feeding_status(hours_since_last_meal):
```

Använd följande testfall för att kontrollera din kod:

```
print(feeding_status(2))   # Output: "Full"
print(feeding_status(4))   # Output: "Full"
print(feeding_status(4.5)) # Output: "Hungry"
print(feeding_status(6))   # Output: "Unhappy"
```

2. (5p)

Ett I en matvarubutik finns det ett lojalitetsprogram som erbjuder rabatter till kunder baserat på deras totala utgifter. Rabattsatserna är följande:
Om en kunds totala utgifter är mindre än $100, finns ingen rabatt.
Om en kunds totala utgifter är mellan $100 och $500 (inkluderande), får de 5% rabatt.
Om en kunds totala utgifter är mellan $501 och $1000 (inkluderande), får de 10% rabatt.
Om en kunds totala utgifter är över $1000, får de 15% rabatt.

Skriv en Python-funktion som heter calculate_discount som tar ett decimaltal total_spending som inmatning och returnerar rabattbeloppet som en kund är berättigad till baserat på deras totala utgifter.

```
def calculate_discount(total_spending):
```

Använd följande testfall för att kontrollera din kod:

```
print(calculate_discount(50))    # Output: 0
print(calculate_discount(150))   # Output: 7.5
print(calculate_discount(600))   # Output: 60.0
print(calculate_discount(1200))  # Output: 180.0
```

3. (5p)

I ett textbaserat spel navigerar spelare genom en labyrint genom att följa en sekvens av riktningar. Varje riktning representeras av ett enskilt tecken: "N" för norr, "S" för söder, "E" för öst och "W" för väst. Spelare startar vid origo (0, 0) och rör sig enligt de givna riktningarna.

Skriv en Python-funktion som heter final_position som tar en sträng directions som inmatning och returnerar spelarens slutposition som en tuple (x, y) efter att ha följt de givna riktningarna.

```
def final_position(directions):
```

Använd följande testfall för att kontrollera din kod:

```
print(final_position("NS"))            # Output: (0, 0)
print(final_position("NESW"))          # Output: (0, 0)
print(final_position("NESNW"))         # Output: (1, 1)
print(final_position("SSWWNEEE"))      # Output: (1, -1)
```

4. (5p)

Du har en lista med strängar. Skriv en funktion som heter remove_short_strings som tar en lista med strängar som inmatning och tar bort alla strängar med en längd mindre än 5 tecken från listan. Funktionen bör returnera den nya genererade listan.

```
def remove_short_strings(string_list):
```

Använd följande testfall för att kontrollera din kod:

```
# Test case 1
words1 = ["apple", "banana", "grape", "kiwi", "pear", "orange"]
print(remove_short_strings(words1))  # Output: ["apple", "banana",
"orange", "grape"]


# Test case 2
words3 = ["python", "java", "c++", "ruby", "swift"]
print(remove_short_strings(words3))  # Output: ["python", "swift"]
```

5. (5p)

Du har en nästlad lista som innehåller information om studenter. Varje inre lista representerar en student och innehåller deras namn (sträng), ålder (heltal) och betyg (decimaltal). Skriv en funktion som heter highest_grade_student som tar en nästlad lista med studentinformation som inmatning och returnerar namnet på studenten med högst betyg.

Om flera studenter har samma högsta betyg bör funktionen returnera namnet på den första studenten med det betyget.

```
def highest_grade_student(student_info):
```

Använd följande testfall för att kontrollera din kod:

```
students = [["Alice", 18, 90.5], ["Bob", 19, 85.0], ["Cathy", 20, 92.3],
["David", 21, 90.5]]
print(highest_grade_student(students))  # Output: "Cathy"

other_students = [["Eva", 22, 80.0], ["Frank", 23, 88.5], ["Grace", 24,
88.5], ["Helen", 25, 82.2]]
print(highest_grade_student(other_students))  # Output: "Frank"
```

6. (5p)

Du har en ordbok som innehåller priserna på olika föremål i en butik. Varje nyckel representerar ett föremåls namn, och det motsvarande värdet är det ordinarie priset (heltal) för föremålet. Butiken har en 20% rabatt på alla föremål. Skriv en funktion som heter total_discount_price som tar ordboken över föremålspris och en lista över föremål som ska säljas som inmatning. Funktionen ska beräkna och returnera det totala rabatterade priset som du kommer att få för de sålda föremålen, med tanke på 20% rabatt.

```
def total_discount_price(item_prices, sold_items):
```

Använd följande testfall för att kontrollera din kod:

```
prices = {'apple': 2, 'banana': 1, 'orange': 1, 'grape': 3, 'kiwi':7}
items_sold1 = ['apple', 'orange', 'kiwi']
items_sold2 = ['banana', 'grape']
items_sold3 = ['orange', 'kiwi']

print(total_discount_price(prices, items_sold1))  # Output: 8.0
print(total_discount_price(prices, items_sold2))  # Output: 3.2
print(total_discount_price(prices, items_sold3))  # Output: 6.4
```