Introduktion och syfte

Målet är att sätta sig in i hur grundläggande sortering fungerar i Python respektive C++, och metoder för detta.

Uppgift

Uppgiften går ut på att implementera:

- En metod för att sortera en kortlek
- Nödvändiga tester

Ni kan välja att implementera i Python eller C++ (eller båda om ni vill).

Python

deck.py

- Er uppgift är att implementera funktionen **insert(card_list, card)**. Denna tar en lista med kort (card_list) och ska infoga ett kort (card) på rätt plats i denna lista. Sen ska listan returneras.
- Metoden kommer anropas 52 gånger, en gång för varje kort i en kortlek. Metoden sort() tar ett kort från den gamla kortleken och lägger det i den nya kortleken med hjälp av insert().

Main-funktionen i den nuvarande koden anropar endast metoden och skriver ut i konsolen. För godkänt räcker det att i main-funktionen lägga in ett fåtal tester som testar er funktion (Deck.insert). Ni kan använda assert som tester. Vad kan vara lämpliga tester? (Storlek, testa positioner etc). För bonuspoäng, se nedan

test_deck.py

Skal för att implementera tester (se bonuspoäng)

C++

Strukturen är uppbyggt som man ofta ser i C++ program med en include directory och en main fil. Det finns en CMake file för att kunna bygga med CMake. (Instruktioner i README-fil)

main.cpp

main funktionen skapar en kortlek (Deck) och kör sorteringsfunktionen och skriver ut kortleken i prompten.

Om ni vill göra minsta möjliga för godkänt, kan ni lägga in assertions i main-funktion som tester. Om ni istället implementerar testfall behöver ni inte det.

deck.cpp

Ni ska implementera den **statiska** funktionen:

```
void Deck::insert(vector<Card> &cardlist, Card card)
```

- Input är en lista med kort (cardlist av typen vector<Card>) och ska infoga ett kort (card) på rätt plats i denna lista. Listan skickas som referens, så inget behöver returneras.
- Metoden kommer anropas 52 gånger, en gång för varje kort i en kortlek. Metoden sort() tar ett kort från den gamla kortleken och lägger det i den nya kortleken med hjälp av insert().

include/deck.h

Här hittar ni definitionen av Card och Deck klassen.

test_deck.cpp

Skal för att implementera tester. Jag har satt upp strukturen för att använda Googletest ramverket. Vill ni hellre använda något annat ramverk (Boost.Test, CTest, CUnit, CppUTest) är det ok. Ni får i så fall sätta upp strukturen för det ramverket.

Bonusuppgifter

Bonusuppgift 1:

(En bonuspoäng inför tentan)

Skriv testfall för alla metoder. Antingen i python eller C++.

Python - test_deck.py

- Ni ska förutom sorteringsalgoritmen även skriva tester för hela kodbasen.
- Implementera lämpliga tester för er sorteringsalgoritm
- - Deck.insert(card_list, card))
- Implementera även tester för de andra metoderna:
- - Card.__init__()
- - Card.__eq__()
- - Card.__lt__()
- - Card.__gt__()
- - Deck.__len__()
- - Deck.sort()

- - Deck.take()
- - Deck.put()
- Skriv en kort motivering för era tester

C++ - test_deck.cpp

För C++ har jag satt upp strukturen med att använda GoogleTest. Ni kan använda annan unit test framework om ni vill (då får ni ändra strukturen själva).

- Deck.insert(vector<Card> &cardlist, Card card)
- Konstruktorn
- Card == -operator
- Card < - operator
- Card > - operator
- Deck.size()
- Deck.sort()
- Deck.take()
- Deck.put(Card card)

Bonusuppgift 2:

(En bonuspoäng inför tentan)

Följande ska implementeras (antingen i C++ eller python):

- Lägg till tre jokrar till kortleken (ska läggas till när kortleken skapas)
- Lägg till minst tre av följande funktioner:
    - sort_by_suit – Sortera så att kort av samma färg (suit) grupperas tillsammans. Inom färgen ska de sorteras i ordning
    - sort_by_value – Sortera korten i nummerordning. Notera: Sorteringen ska alltid bli samma, så ni måste ta hänsyn även till färgen (suit)
    - pick_by_random – Välj ut (och plocka bort) ett kort slumpmässigt ur leken.
    - deal(int n) – Dela ut kortleken till n antal personer. Här får ni tänka efter hur ni ska implementera denna, så att ni får jämnt antal korthögar. (Kan vara separat / statisk funktion)
    - remove_duplicates – Leta upp och ta bort kort som är duplicerade
    - remove_jokers – Ta bort jokrarna

Redovisning

Källkoden ska vara pushad till ett eget repository på Codeberg eller GitHub. Skriv namn i README.md i repots main-branch (master).

Inlämningen sker genom att en länk till gitrepot skickas in i Omniway.