Информатика

В задачата можете да използвате наготово класа std::vector от <vector>.

Маймуната Зимбу решил, че е крайно време да започне сериозен бизнес. Навремето, когато пристигнал в България, докато го карали към зоопарка, шофьорът минал прекалено близо до няколко университета. Така докато стигнат, Зимбу вече бил собственик на няколко бакалавърски дипломи и на две магистърски. Когато пристигнали в зоопарка, оттам му казали, че е прекалено квалифициран и не го пуснали вътре. Така той се оказал в сложна ситуация -- трябвало сам да си изкарва прехраната.

Зимбу разгледал дипломите си и особено му харесала една, издадена от "Вселенския университет за висши науки и съвършенство във всички области" от Горно Нанадолнище. На нея пишело: "ИТ специалист!" (с удивителна). И понеже сега "компютрите са модерни", Зимбу решил да продава компютри. Веднага си регистрирал фирма (ЕТ "Стратиджик инфлуенсинг енд трендсетинг – Зимбу Маймуната") и започнал дейността си в един гараж. Но Зимбу не бил вчерашен. Още с пристигането си тук, той бил разбрал, че не иска да работи и му се струвало съвсем естествено да не прави нищо, а да получава по много. Затова той иска да автоматизира дейността си и някой да му напише програма (разбира се, безплатно – "Тия програмисти какво само искат!"). С нея хората сами ще могат да си направят конфигурация и да си я продадат сами на себе си. Вашата задача е да помогнете на Зимбу.

А) (2 точки) Един компютър се състои от един или повече компоненти. Всички компоненти са immutable обекти. След като веднъж бъдат създадени, свойствата им не могат да се променят. Всеки компонент да има:

етикет (label), който е std::string обект.

функция double price() const, която връща цената на компонента. Тя ще се имплементира от всеки конкретен компонент и ще пресмята цената по някакво правило.

функция void output(std::ostream& out) const, която извежда информация за компонента – неговото име и цена.

Програмата ви трябва да поддържа два конкретни компонента – процесор (cpu) и памет (memory).

Процесорът има брой ядра (cores) и честота в MHz (clock speed). И двете са числа от тип unsigned short. Броят на ядрата може да е между 1 и 8. (Зимбу много държи да не се мине и да не продаде някое ядро в повече). За честотата няма ограничение ("Нека има, да се радват клиентите!" казва Зимбу). Цената на процесора се смята като броя на ядрата, умножен по 29.99 лева.

Паметта има капацитет (capacity) в гигабайти (unsigned short). Трябва да е число между 1 и 10'000 (Зимбу лъже в обявите, защото клиентите му са лековерни). Цената на паметта е 89.99 лева на гигабайт.

Ако се опита да се създаде компонент с некоректно подадени параметри, да се хвърли изключение.

Реализирайте подходяща йерархия, за да представите горе-описаната ситуация. Проектирайте йерархията по всички добри практики, които познавате, като имате предвид, че Зимбу в бъдеще ще добавя и други видове компоненти. Например помислете какъв трябва да е деструкторът, кои операции да са pure virtual и кои да изнесете в базовия клас и т.н.

Б) (2 точки) Напишете factory функция component\* create\_component(). Тя трябва да попита потребителя какъв тип компонент иска да създаде – процесор или памет. След това тя въвежда съответните му характеристики, създава динамично (със new) обект от съответния тип и го връща. Ако потребителят въведе невалиден тип или създаването на обект е неуспешно, да се върне nullptr.

В) (4 точки) Напишете клас configuration представящ конфигурация. Класът трябва да може да съхранява в себе си един или повече компоненти. Няма ограничение за техния брой – сега имаме само два вида, но в бъдеще Зимбу ще добави и още.

Класът да има следните член-функции:

double price() const – връща цената на конфигурацията. Пресмята се като сума от цените на всички компоненти.

std::size\_t size() const – връща броя на компонентите в конфигурацията.

За класа предефинирайте следните оператори:

operator[], който позволява да се достъпи един от компонентите на конфигурацията. Операторът да приема стойност от тип std::size\_t – индекс на компонент (число между 0 и configuration::size()-1). Направете проверка дали подаденият индекс е коректен и ако това не е така, хвърлете изключение. Операторът трябва да е константна функция и да връща const reference. През него не трябва да може да се променят компонентите.

operator<<, който извежда информация за конфигурацията в std::ostream. При извеждането най-напред да се изведат един по един всички компоненти, използвайки техните функции output. Накрая, да се изведе и още един ред с общата цена на конфигурацията.

Изграждането на конфигурацията ще бъде сложно и затова не трябва да може всеки да го прави. За целта:

Дефинирайте default конструктора като private. Той да създава празна конфигурация, в която няма компоненти.

Дефинирайте private функция void insert(const component\* c), която добавя нов компонент към конфигурацията. Функцията директно да съхранява указателя, без да прави копие на подадения компонент.

Деструкторът на класа трябва да изтрива всички съхранени в него обекти.

Предотвратете копирането на конфигурации, като направите копиращия конструктор и копиращото присвояване или да са private и без дефиниция, или да са изтрити функции.

Относно insert имаме и още едно изискване. В една конфигурация не може да има повече от един компонент от даден тип. Когато добавяте нов компонент, изполвайте RTTI/typeid, за да проверите дали новият компонент няма същия тип като някой, който вече е добавен. Ако това е така, компонентът да не се добавя, а да се хвърли изключение.

Г) (2 точки) Напишете функция create\_configuration. Тя трябва да създава и да връща configuration обект. Направете я friend за класа configuration.

Функцията да създава динамично нов configuration обект (със new), да въвежда за него един или повече компоненти от потребителя (чрез create\_component), да ги добавя в конфигурацията (чрез configuration::insert) и накрая да връща готовия обект.

Ако по някаква причина създаването не успее, върнете nullptr.

Д) (2 точки) Напишете програма, която използва горните класове, за да направи следното:

Позволява на потребителя да въведе каквито пожелае компоненти

Изгражда по тях конфигурация

Извежда на екрана избраните от потребителя компоненти и цената на конфигурацията.

Същата информация да се изведе и в текстов файл с име посочено от потребителя.