4. Наследяване.

I. Наследяване. Конструктори и деструктори на производни класове.

Когато един клас наследява друг се използва общата форма:

```
class име-производен-клас:тип-достъп име-базов-клас {
          //...
};
```

Тук *тип-достъп* е една от трите ключови думи: **public**, **private** или **protected**. Ако базовият клас е деклариран като *public* в производния клас, всички *public*, *private* и *protected* компоненти на базовия клас се наследяват съответно като *public*, *private* и *protected* компоненти на производния клас.

Ако е необходимо да се предаде аргумент на конструктора на базовия клас, то всички необходими аргументи за базовия и за производния клас се предават на конструктора на производния клас. Като се използва разширената форма на декларация на конструктора на производния клас, могат да се предадат съответните аргументи и към базовия клас:

```
производен-конструктор(списък аргументи):базов-клас(списък аргументи) {
    //тяло на конструктора на производния клас
}
```

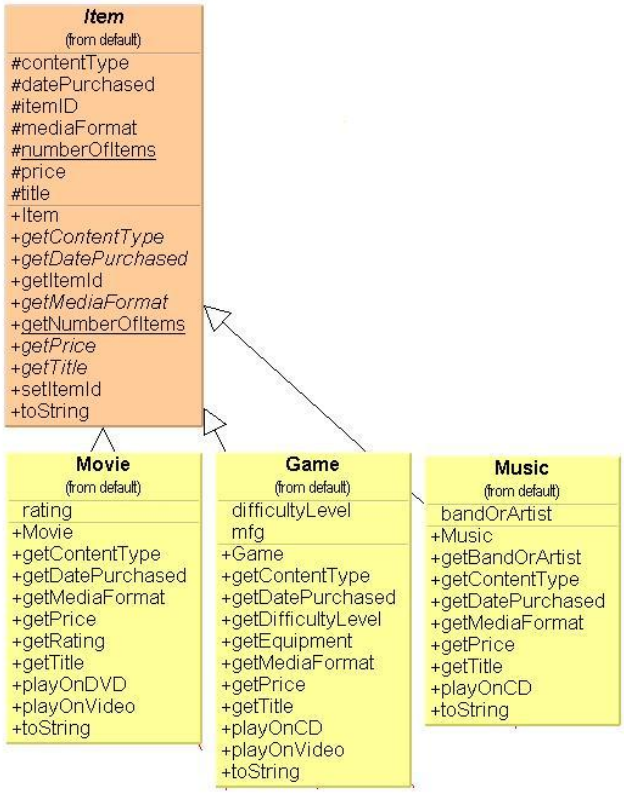
Задача 1. В следната програма е дефиниран базов клас за превозно средство – **vehicle**, и два производни класа – **car** и **truck**. Да се създадат конструктори **car()** и **truck()**; **car()** да инициализира член-променливата **passengers**, а **truck()** да инициализира член-променливата **loadlimit**. Нека **car()** и **truck()** да предават подходящи аргументи на конструктора **vehicle()**. Да се дефинират обекти от класовете **car** и **truck** и да се изведат дефинираните обекти.

```
#include<iostream.h>
//Базов клас за различни превозни средства
class vehicle {
       int num_wheels;
       int range;
    public:
       vehicle(int w, int r)  {  num_wheels = w; range = r; }
       void showv() {
         cout << ”Wheels:” << num_wheels << ’\n’;
         cout << ”Range:” << range << ’\n’;
       }
};
class car : public vehicle {
       int passengers;
    public:
       car(int p, int w, int r) : vehicle(w, r)
       {
         passengers = p;
       }
       void show()
       { showv();
         cout << ”Passengers:” << passengers << ’\n’;
       }
};
class truck : public vehicle {
       int loadlimit;
    public:
       truck(int l, int w, int r) : vehicle(w, r) { loadlimit = l; }
       void show()
       { showv();
```

```
        cout << ”Loadlimit:” << loadlimit << ’\n’;
      }
};
int main()
{
  car c(5, 4, 500);
  truck t(30000, 12, 1200);
  cout << ”Car:\n”;
  c.show();
  cout << ”\nTruck:\n”;
  t.show();
  return 0;
}
```

Задачи за самостоятелна работа:

Приложение за управление на лична библиотека от цифрова медия (Media Collection Tool):
Медия това е:DVD, CD, видео.
Съдържанието на медията може да е: музика, филм , игри
Имплементирайте следните класове и нивата на наследяване:



Item - базов клас, който описва една медия: заглавие, цена, номер на медията (уникален), формат на медията, дата на закупуването. Конструктор за инициализация, член-функции за получаване на достъп до характеристиките на класа и член – функция за извеждане на съобщение от вида:
„ <<Вид медия>> с заглавие <<заглавие>> е била за купена на цена <<цена>> на дата <<дата>>.”
Movie – наследява и разширява базовия клас с променливата рейтинг на филма.
Game – наследява и разширява базовия клас с променливата ниво на трудност на играта.
Music – наследява и разширява базовия клас с променливата изпълнители на музиката.