**FILE PROGRAM.CS**

using System;

namespace hw1

{

public class Program

{

static void Main(string[] args)

{

try

{

Product CPU = new("Intel I9-9900K", 449.90, 0.10);

Product GPU = new("nVidia GeForce GTX 3090 Ti", 500, 1);

Product RAM = new("Kingston 3200 16 GB", 200, 0.10);

Product SSD = new("Samsung EVO 980 m.2 1TB", 600, 0.1);

Console.WriteLine(Check.PrintProduct(CPU));

Console.WriteLine(Check.PrintProduct(SSD));

Buy PCPurchase = new(CPU, GPU, RAM, SSD);

Buy ShopPurchase = new(CPU, -10);

Console.WriteLine("\nPC purchase:\r\b" + Check.PrintBuy(PCPurchase));

}

catch (ArgumentException ex)

{

Console.WriteLine("\nError occured:\n" + ex.Message + "\nExiting programm");

return;

}

}

}

}

**FILE PRODUCT.CS**

using System;

namespace hw1

{

public class Product

{

private string \_name;

private double \_price;

private double \_weight;

public string Name

{

get => \_name;

set

{

if (String.Compare(value, "") != 0)

{

\_name = value;

}

else

{

throw new ArgumentException("Name of product can't be empty");

}

}

}

public double Price

{

get => \_price;

set

{

if (value > 0)

{

\_price = value;

}

else

{

throw new ArgumentException("Price must be a positive number");

}

}

}

public double Weight

{

get => \_weight;

set

{

if (value > 0)

{

\_weight = value;

}

else

{

throw new ArgumentException("Weight must be a positive number");

}

}

}

public Product(string name, double price, double weight)

{

Name = name;

Price = price;

Weight = weight;

}

}

}

**FILE BUY.CS**

using System;

using System.Collections.Generic;

using System.Linq;

namespace hw1

{

public class Buy

{

private List<Product> \_purchaseList;

private int \_amount;

private double \_totalPrice;

private double \_totalWeight;

public Buy(Product purchase, int amount)

{

\_purchaseList = new List<Product>();

\_purchaseList.Add(purchase);

Amount = amount;

\_totalPrice = \_amount \* this.\_purchaseList[0].Price;

\_totalWeight = \_amount \* this.\_purchaseList[0].Weight;

}

public Buy(params Product[] products)

{

\_purchaseList = new List<Product>();

Amount = products.Length;

for (int i = 0; i < products.Length; ++i)

{

if (products[i] == null)

{

throw new ArgumentException("Invalid agument");

}

else

{

\_purchaseList.Add(products[i]);

}

}

\_totalPrice = \_purchaseList.Sum(product => product.Price);

\_totalWeight = \_purchaseList.Sum(product => product.Weight);

}

public List<Product> PurchaseList

{

get => \_purchaseList;

}

public int Amount

{

get => \_amount;

set

{

if (value > 0)

{

\_amount = value;

}

else

{

throw new ArgumentException("Amount must be a positive number");

}

}

}

public double TotalPrice { get => \_totalPrice; }

public double TotalWeight { get => \_totalWeight; }

}

}

**FILE CHECK.CS**

using System.Text;

namespace hw1

{

public class Check

{

public static string PrintBuy(Buy buy)

{

var result = new StringBuilder();

for (int i = 0; i < buy.PurchaseList.Count; ++i)

{

result.Append("\n" + PrintProduct(buy.PurchaseList[i]));

}

result.Append($"\n\nAmount: {buy.Amount.ToString() } pcs");

result.Append($"\nTotal Price: {buy.TotalPrice.ToString("$0.00")}");

result.Append($"\nTotal Weight: {buy.TotalWeight.ToString("0.00 kg")}\n");

return result.ToString();

}

public static string PrintProduct(Product product)

{

var result = new StringBuilder();

result.Append($"\nName: {product.Name}");

result.Append($"\nPrice: {product.Price.ToString("$0.00")}");

result.Append($"\nWeight: {product.Weight.ToString("0.00 kg")}");

return result.ToString();

}

}

}