**8 Коллекции, классы-прототипы, обобщения**

Задание 1. Очередь заявок на обслуживание (Queue)

o Класс ServiceRequest с Id, CustomerName, RequestType.

o Класс ServiceRequestManager управляет Queue&lt;ServiceRequest&gt;.

Листинг задачи:

using System.Collections;

namespace Task1

{

public class ServiceRequestManager

{

private readonly Queue \_requestQueue = new Queue();

public void AddRequest(ServiceRequest request)

{

\_requestQueue.Enqueue(request);

Console.WriteLine($"Заявка от {request.CustomerName} добавлена.");

}

public ServiceRequest ProcessNextRequest()

{

if (\_requestQueue.Count == 0)

{

Console.WriteLine("Очередь пуста.");

return null;

}

ServiceRequest request = (ServiceRequest)\_requestQueue.Dequeue();

Console.WriteLine($"Обработка заявки от {request.CustomerName}.");

return request;

}

public ArrayList FindRequestsByType(string requestType)

{

ArrayList result = new ArrayList();

foreach (var item in \_requestQueue)

{

ServiceRequest request = (ServiceRequest)item;

if (request.RequestType == requestType)

{

result.Add(request);

}

}

return result;

}

public void DisplayAllRequests()

{

foreach (var item in \_requestQueue)

{

ServiceRequest request = (ServiceRequest)item;

Console.WriteLine(request);

}

}

public ArrayList GetAllRequestsSortedByDate()

{

ArrayList requests = new ArrayList(\_requestQueue);

requests.Sort(new ServiceRequestDateComparer());

return requests;

}

}

}

using System.Collections;

namespace Task1

{

public class ServiceRequestDateComparer : IComparer

{

public int Compare(object x, object y)

{

if (x is ServiceRequest requestX && y is ServiceRequest requestY)

{

return requestX.CreatedDate.CompareTo(requestY.CreatedDate);

}

return 0;

}

}

}

namespace Task1

{

public class ServiceRequest

{

public Guid Id { get; }

public string CustomerName { get; set; }

public string RequestType { get; set; }

public DateTime CreatedDate { get; }

public ServiceRequest(string customerName, string requestType)

{

Id = Guid.NewGuid();

CustomerName = customerName;

RequestType = requestType;

CreatedDate = DateTime.Now;

}

public override string ToString()

{

return $"Id: {Id}, Customer: {CustomerName}, Type: {RequestType}, Created: {CreatedDate}";

}

}

} using System.Collections;

using Task1;

class Program

{

static void Main(string[] args)

{

var requestManager = new ServiceRequestManager();

requestManager.AddRequest(new ServiceRequest("Иван", "Тип1"));

requestManager.AddRequest(new ServiceRequest("Петр", "Тип2"));

requestManager.AddRequest(new ServiceRequest("Сидор", "Тип1"));

Console.WriteLine("Все заявки (отсортированные по дате):");

ArrayList sortedRequests = requestManager.GetAllRequestsSortedByDate();

foreach (ServiceRequest r in sortedRequests)

{

Console.WriteLine(r);

}

Console.WriteLine("\nЗаявки типа Тип1:");

ArrayList requests = requestManager.FindRequestsByType("Тип1");

foreach (ServiceRequest r in requests)

{

Console.WriteLine(r);

}

Console.WriteLine("\nОбработка заявок:");

requestManager.ProcessNextRequest();

requestManager.ProcessNextRequest();

requestManager.ProcessNextRequest();

requestManager.ProcessNextRequest(); // проверка пустой очереди

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Заявка от Ивана, заявка от Петра, заявка от Дмитрия | Заявки добавлены, отсортированы и обработаны |

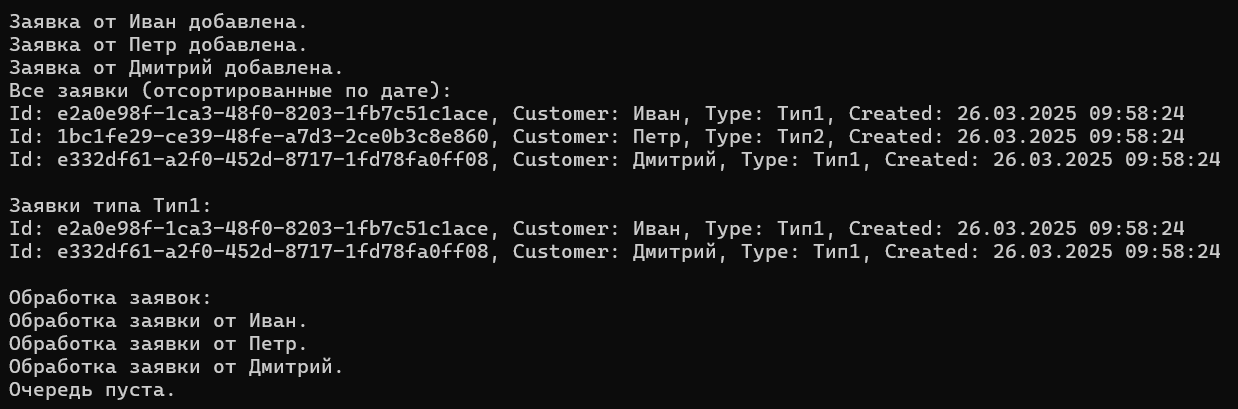
Анализ результатов:

Рисунок 1.1 – Результат работы программы