Questão 1 Nativa - Main

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Nativa
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int escolha;
            List<int> lista = new List<int>();
            int horas, pos;
            do
            {
                Console.WriteLine("\n1) Inserir um tempo no início da lista " +
                    "\n2) Inserir um tempo no final da lista" +
                    "\n3) Inserir um tempo numa posição específica da lista(O usuário deve
informar a posição e o tempo a ser inserido)" +
                    "\n4) Remover o primeiro tempo da lista(Imprimir o tempo removido)" +
                    "\n5) Remover o último tempo da lista(Imprimir o tempo removido)" +
                    "\n6) Remover um tempo de uma posição específica na lista(O usuário
deve informar a posição do tempo a ser removido. Imprimir o tempo removido)" +
                    "\n7) Remover um tempo específico da lista(O usuário deve informar o
tempo a ser removido)." +
                    "\n8) Pesquisar quantas vezes um determinado tempo consta na lista(O
usuário deve informar o tempo a ser pesquisado)." +
                    "\n9) Mostrar todos os tempos da lista" +
                    "\n10) Encerrar o programa");
                escolha = int.Parse(Console.ReadLine());
                switch (escolha)
                {
                    case 1:
                        Console.WriteLine("Insira as horas: ");
                        horas = int.Parse(Console.ReadLine());
                        lista.Insert(0, horas);
                        break;
                    case 2:
                        Console.WriteLine("Insira as horas:");
                        horas = int.Parse(Console.ReadLine());
```

```
                    lista.Add(horas);
                    break;
                case 3:
                    Console.WriteLine("Insira as horas que deseja inserir e em qual
posição");
                    horas = int.Parse(Console.ReadLine());
                    pos = int.Parse(Console.ReadLine());
                    lista.Insert(pos, horas);
                    break;
                case 4:
                    Console.WriteLine("{0} removido com sucesso.", lista[0]);
                    lista.RemoveAt(0);
                    break;
                case 5:
                    Console.WriteLine("{0} removido com sucesso.", lista[lista.Count - 1]);
                    lista.RemoveAt(lista.Count - 1);
                    break;
                case 6:
                    Console.WriteLine("Qual posição deseja remover: ");
                    pos = int.Parse(Console.ReadLine());
                    Console.WriteLine("{0} removido com sucesso.", lista[pos]);
                    lista.RemoveAt(pos);
                    break;
                case 7:
                    Console.WriteLine("Insira o tempo que deseja remover: ");
                    horas = int.Parse(Console.ReadLine());
                    lista.Remove(horas);
                    break;
                case 8:
                    Console.WriteLine("Insira o número que deseja verificar repetições:
");
                    horas = int.Parse(Console.ReadLine());
                    int[] lista2 = lista.ToArray();
                    int repet = 0;
                    for (int i = 0; i < lista.Count - 1; i++)
                    {
                        if (lista2[i] == horas)
                        {
                            repet++;
                        }
                    }
                    Console.WriteLine("O elemento se repete {0} vezes", repet);
                    break;
                case 9:
```

```
                for (int i = 0; i < lista.Count - 1; i++)
                {
                    Console.Write(lista[i] + " / ");
                }
                break;
            default:
                break;
        }
    } while (escolha != 10);
    Console.ReadLine();
  }
 }
}
```

Questão 1 Celula - Celula

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11CelulaQUestão1
{
  internal class Celula
  {
      private Inteiro valor;

      public Inteiro Valor
      {
          get { return valor; }
          set { valor = value; }
      }
      private Celula proximo;

      public Celula Proximo
      {
          get { return proximo; }
          set { proximo = value; }
      }
      public Celula()
      {
          valor = new Inteiro();
```

```
            proximo = null;
        }
        public Celula(Inteiro valor)
        {
            this.valor = valor;
        }
    }
}
```

Questão 1 Celula - Inteiro

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11CelulaQUestão1
{
    internal class Inteiro
    {
        private int item;

        public int Item
        {
            get { return item; }
            set { item = value; }
        }
        public Inteiro()
        {
            item = 0;
        }
        public Inteiro(int item)
        {
            this.item = item;
        }
        public void Imprimir()
        {
            Console.WriteLine("Item: {0}", item);
        }
    }
}
```

Questão 1 Celula - Lista

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Remoting.Messaging;
using System.Text;
using System.Threading.Tasks;

namespace Lista11CelulaQUestão1
{
    internal class Lista
    {
        private Celula primeiro, ultimo;
        private int tamanho;
        public int Tamanho
        {
            get { return tamanho; }
        }
        public Lista()
        {
            Celula sentinela = new Celula();
            primeiro = sentinela;
            ultimo = sentinela;
            tamanho = 0;
        }
        public bool Vazia()
        {
            if (primeiro == ultimo)
            {
                return true;
            }
            else
            {
                return false;
            }
        }
        public void Inserir(Inteiro valor, int pos)
        {
            if (pos >= 0 && pos <= tamanho)
            {
                Celula novaCelula = new Celula(valor);
                Celula anterior = primeiro;
```

```
            Celula proximaCelula;
            for (int i = 0; i < pos; i++)
            {
                anterior = anterior.Proximo;
            }
            proximaCelula = anterior.Proximo;
            anterior.Proximo = novaCelula;
            novaCelula.Proximo = proximaCelula;
            if (pos == tamanho)
            {
                ultimo = novaCelula;
            }
            tamanho++;
        }
        else
        {
            throw new Exception("Posição inválida");
        }
    }
    public Inteiro Remover(int pos)
    {
        if (!Vazia())
        {
            if (pos >= 0 && pos < tamanho)
            {
                Celula anterior = primeiro;
                for (int i = 0; i < pos; i++)
                {
                    anterior = anterior.Proximo;
                }
                Celula celulaRemovida = anterior.Proximo;
                Celula proximaCelula = celulaRemovida.Proximo;
                anterior.Proximo = proximaCelula;
                celulaRemovida.Proximo = null;
                if (celulaRemovida == ultimo)
                {
                    ultimo = anterior;
                }
                tamanho--;
                return celulaRemovida.Valor;
            }
            else
            {
                throw new Exception("Posição inválida");
```

```
        }
    }
    else
    {
        throw new Exception("Vazia!");
    }
}
public Inteiro RemoverDireto(Inteiro horas)
{
    if (!Vazia())
    {
        Celula anterior = primeiro;
        for (int i = 0; anterior.Proximo.Valor != horas; i++)
        {
            anterior = anterior.Proximo;
        }
        Celula removida = anterior.Proximo;
        Celula proxima = removida.Proximo;
        removida.Proximo = null;
        if (ultimo == removida)
        {
            ultimo = anterior;
        }
        tamanho--;
        return removida.Valor;
    }
    else
    {
        throw new Exception("Fila Vazia!");
    }
}
public int Contar(Inteiro horas)
{
    if (!Vazia())
    {
        Celula aux = primeiro;
        int cont = 0;
        while(aux != null)
        {
            if(aux.Valor == horas)
            {
                cont++;
            }
        }
```

```
            return cont;
        }
        else
        {
            throw new Exception("Lista Vazia!");
        }

    }
    public void Imprimir()
    {
        if (!Vazia())
        {
            Celula aux = primeiro.Proximo;
            while (aux != null)
            {
                aux.Valor.Imprimir();
                aux = aux.Proximo;
            }
        }
        else
        {
            throw new Exception("Lista vazia!");
        }
    }
  }
}
```

Questão 1 Celula - Main

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11CelulaQUestão1
{
  internal class Program
  {
    static void Main(string[] args)
    {
        int escolha;
```

```
        Lista lista = new Lista();
        int horas, pos;
        Inteiro horasInteiro, removido;
        do
        {
            Console.WriteLine("\n1) Inserir um tempo no início da lista " +
                "\n2) Inserir um tempo no final da lista" +
                "\n3) Inserir um tempo numa posição específica da lista(O usuário deve
informar a posição e o tempo a ser inserido)" +
                "\n4) Remover o primeiro tempo da lista(Imprimir o tempo removido)" +
                "\n5) Remover o último tempo da lista(Imprimir o tempo removido)" +
                "\n6) Remover um tempo de uma posição específica na lista(O usuário
deve informar a posição do tempo a ser removido. Imprimir o tempo removido)" +
                "\n7) Remover um tempo específico da lista(O usuário deve informar o
tempo a ser removido)." +
                "\n8) Pesquisar quantas vezes um determinado tempo consta na lista(O
usuário deve informar o tempo a ser pesquisado)." +
                "\n9) Mostrar todos os tempos da lista" +
                "\n10) Encerrar o programa");
            escolha = int.Parse(Console.ReadLine());
            switch (escolha)
            {
                case 1:
                    Console.WriteLine("Insira as horas: ");
                    horas = int.Parse(Console.ReadLine());
                    horasInteiro = new Inteiro(horas);
                    lista.Inserir(horasInteiro, 0);
                    break;
                case 2:
                    Console.WriteLine("Insira as horas:");
                    horas = int.Parse(Console.ReadLine());
                    horasInteiro = new Inteiro(horas);
                    lista.Inserir(horasInteiro, lista.Tamanho);
                    break;
                case 3:
                    Console.WriteLine("Insira as horas que deseja inserir e em qual
posição");
                    horas = int.Parse(Console.ReadLine());
                    horasInteiro = new Inteiro(horas);
                    pos = int.Parse(Console.ReadLine());
                    lista.Inserir(horasInteiro, pos);
                    break;
                case 4:
                    removido = lista.Remover(0);
```

```
                    Console.WriteLine("{0} removido com sucesso.", removido.Item);
                    break;
                case 5:
                    removido = lista.Remover(lista.Tamanho);
                    Console.WriteLine("{0} removido com sucesso.", removido.Item);
                    break;
                case 6:
                    Console.WriteLine("Qual posição deseja remover: ");
                    pos = int.Parse(Console.ReadLine());
                    removido = lista.Remover(pos);
                    Console.WriteLine("{0} removido com sucesso.", removido.Item);
                    break;
                case 7:
                    Console.WriteLine("Insira o tempo que deseja remover: ");
                    horas = int.Parse(Console.ReadLine());
                    horasInteiro = new Inteiro(horas);
                    lista.RemoverDireto(horasInteiro);
                    break;
                case 8:
                    Console.WriteLine("Insira o número que deseja verificar repetições:
");
                    horas = int.Parse(Console.ReadLine());
                    horasInteiro = new Inteiro(horas);
                    Console.WriteLine("O elemento se repete {0} vezes",
lista.Contar(horasInteiro));
                    break;
                case 9:
                    lista.Imprimir();
                    break;
                default:
                    break;
            }
        } while (escolha != 10);
        Console.ReadLine();
    }
  }
}
```

Questão 1 Vetor - Inteiro

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11Questão1Vetor
{
    internal class Inteiro
    {
        private int item;

        public int Item
        {
            get { return item; }
            set { item = value; }
        }
        public Inteiro()
        {
            item = 0;
        }
        public Inteiro(int item)
        {
            this.item = item;
        }
        public void Imprimir()
        {
            Console.WriteLine("Item: {0}", item);
        }
    }
}
```

Questão 1 Vetor - Vetor

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11Questão1Vetor
{
    internal class Lista
    {
```

```
private int primeiro;
private int ultimo;
private int tamanho;
private Inteiro[] lista;
public Lista(int size)
{
    primeiro = 0;
    ultimo = 0;
    tamanho = 0;
    lista = new Inteiro[size];
}
public int Count()
{
    return tamanho;
}
public bool Cheia()
{
    if (ultimo == lista.Length)
    {
        return true;
    }
    else
    {
        return false;
    }
}
public bool Vazia()
{
    if (primeiro == ultimo)
    {
        return true;
    }
    else
    {
        return false;
    }
}
public void Inserir(int pos, Inteiro valor)
{
    if (!Cheia())
    {
        if (pos >= 0 && pos <= tamanho)
        {
            for (int i = ultimo; i > pos; i--)
```

```
            {
                lista[i] = lista[i - 1];
            }
            lista[pos] = valor;
            tamanho++;
            ultimo++;
        }
    }
    else
    {
        throw new Exception("Lista cheia");
    }
}
public int VerificarPos(Inteiro horas)
{
    if (!Vazia())
    {
        for(int i = primeiro; i < tamanho, i++)
        {
            if (lista[i] == horas)
            {
                return i;
            }
        }
        throw new Exception("Elemento Inválido");
    }
    else
    {
        throw new Exception("Lista Vazia!");
    }
}
public Inteiro Remover(int pos)
{
    if (!Vazia())
    {
        if (pos >= 0 && pos < tamanho)
        {
            Inteiro removido = lista[pos];
            tamanho--;
            for (int i = pos; i < tamanho; i++)
            {
                lista[i] = lista[i + 1];
            }
            ultimo--;
```

```
            return removido;
        }
        else
        {
            throw new Exception("Inválido");
        }
    }
    else
    {
        throw new Exception("Lista vazia");
    }
}
public void RemoverHoras(Inteiro horas)
{
    if (!Vazia())
    {
        int pos = VerificarPos(horas);
        Inteiro removido = lista[pos];
        tamanho--;
        for (int i = pos; i < tamanho; i++)
        {
            lista[i] = lista[i + 1];
        }
        ultimo--;
    }
    else
    {
        throw new Exception("Lista vazia");
    }
}
public void Imprimir()
{
    if (!Vazia())
    {
        for(int i = primeiro; i < ultimo; i++)
        {
            Console.Write(lista[i] + " ");
        }
    }
    else
    {
        throw new Exception("Lista Vazia!");
    }
```

```
        }
        public int Repetir(Inteiro horas)
        {
            if (!Vazia())
            {
                int cont = 0;
                for(int i = primeiro; i < ultimo; i++)
                {
                    if (lista[i] == horas) {
                        cont++;
                    }
                }
            }
        }
    }
}
```

Questão 1 Vetor - Main

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11Questão1Vetor
{
    internal class Program
    {
        static void Main(string[] args)
        {
            int escolha;
            Lista lista = new Lista(20);
            Inteiro horasInteiro;
            int horas, pos, removido;
            do
            {
                Console.WriteLine("\n1) Inserir um tempo no início da lista " +
                    "\n2) Inserir um tempo no final da lista" +
                    "\n3) Inserir um tempo numa posição específica da lista(O usuário deve
informar a posição e o tempo a ser inserido)" +
                    "\n4) Remover o primeiro tempo da lista(Imprimir o tempo removido)" +
                    "\n5) Remover o último tempo da lista(Imprimir o tempo removido)" +
```

```
            "\n6) Remover um tempo de uma posição específica na lista(O usuário
deve informar a posição do tempo a ser removido. Imprimir o tempo removido)" +
            "\n7) Remover um tempo específico da lista(O usuário deve informar o
tempo a ser removido)." +
            "\n8) Pesquisar quantas vezes um determinado tempo consta na lista(O
usuário deve informar o tempo a ser pesquisado)." +
            "\n9) Mostrar todos os tempos da lista" +
            "\n10) Encerrar o programa");
        escolha = int.Parse(Console.ReadLine());
        switch (escolha)
        {
            case 1:
                Console.WriteLine("Insira as horas: ");
                horas = int.Parse(Console.ReadLine());
                horasInteiro = new Inteiro(horas);
                lista.Inserir(0, horasInteiro);
                break;
            case 2:
                Console.WriteLine("Insira as horas:");
                horas = int.Parse(Console.ReadLine());
                horasInteiro = new Inteiro(horas);
                lista.Inserir(lista.Count(),horasInteiro);
                break;
            case 3:
                Console.WriteLine("Insira as horas que deseja inserir e em qual
posição");
                horas = int.Parse(Console.ReadLine());
                horasInteiro = new Inteiro(horas);
                pos = int.Parse(Console.ReadLine());
                lista.Inserir(pos, horasInteiro);
                break;
            case 4:
                removido = lista.Remover(0).Item;
                Console.WriteLine("{0} removido com sucesso.", removido);
                break;
            case 5:
                removido = lista.Remover(lista.Count()).Item;
                Console.WriteLine("{0} removido com sucesso.", removido);

                break;
            case 6:
                Console.WriteLine("Qual posição deseja remover: ");
                pos = int.Parse(Console.ReadLine());
                removido = lista.Remover(pos).Item;
```

```
                Console.WriteLine("{0} removido com sucesso.", removido);

                break;
            case 7:
                Console.WriteLine("Insira o tempo que deseja remover: ");
                horas = int.Parse(Console.ReadLine());
                horasInteiro = new Inteiro(horas);
                lista.RemoverHoras(horasInteiro);
                break;
            case 8:
                Console.WriteLine("Insira o tempo que deseja remover: ");
                horas = int.Parse(Console.ReadLine());
                horasInteiro = new Inteiro(horas);
                int repet = lista.Repetir(horasInteiro);
                Console.WriteLine("O elemento se repete {0} vezes", repet);
                break;
            case 9:
                lista.Imprimir();
                break;
            default:
                break;
        }
      } while (escolha != 10);
      Console.ReadLine();
    }
  }
}
```

Questão 2 Nativa - Inteiro

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11Questão2
{
  internal class Site
  {
    private string nome;

    public string Nome
    {
```

```
            get { return nome; }
            set { nome = value; }
        }
        private string link;

        public string Link
        {
            get { return link; }
            set { link = value; }
        }

        public Site()
        {
            nome = null;
            link = null;
        }
        public Site(string nome, string link)
        {
            this.nome = nome;
            this.link = link;
        }
        public void Imprimir()
        {
            Console.WriteLine("Site: {0}:{1}", nome, link);
        }
    }
}
```

Questão 2 Nativa - Main

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lista11Questão2
{
    internal class Program
    {
        static void Main(string[] args)
        {
            List<Site> lista = new List<Site>();
```

```
        int escolha, pos;
        string nome, link, removido;
        Site site;
        do
        {
            Console.WriteLine("1) Inserir um Site no início da lista\r\n2) Inserir um Site
no final da lista\r\n3) Inserir um Site numa posição específica da lista\r\n4) Remover
o primeiro Site da lista (Imprimir o nome do site removido)\r\n5) Remover o último
Site da lista (Imprimir o nome do site removido)\r\n6) Remover um Site de uma
posição específica da lista (Imprimir o nome do site removido)\r\n7) Mostrar o nome
e o link de todos os sites da lista\r\n8) Encerrar o programa");
            escolha = int.Parse(Console.ReadLine());
            switch (escolha)
            {
                case 1:
                    Console.WriteLine("Insira o nome e o link do site");
                    nome = Console.ReadLine();
                    link = Console.ReadLine();
                    site = new Site(nome, link);
                    lista.Insert(0, site);
                    break;
                case 2:
                    Console.WriteLine("Insira o nome e o link do site");
                    nome = Console.ReadLine();
                    link = Console.ReadLine();
                    site = new Site(nome, link);
                    lista.Add(site);
                    break;
                case 3:
                    Console.WriteLine("Insira o nome, o link do site e a posição que
deseja inserir");
                    nome = Console.ReadLine();
                    link = Console.ReadLine();
                    pos = int.Parse(Console.ReadLine());
                    site = new Site(nome, link);
                    lista.Insert(pos, site);
                    break;
                case 4:
                    removido = lista[0].Nome;
                    lista.RemoveAt(0);
                    Console.WriteLine("Item {0} removido com sucesso", removido);
                    break;
                case 5:
                    removido = lista[lista.Count - 1].Nome;
```

```
                lista.RemoveAt(lista.Count - 1);
                Console.WriteLine("Item {0} removido com sucesso", removido);
                break;
            case 6:
                Console.WriteLine("Insira a posição que deseja remover: ");
                pos = int.Parse(Console.ReadLine());
                removido = lista[pos].Nome;
                lista.RemoveAt(pos);
                Console.WriteLine("Item {0} removido com sucesso", removido);
                break;
            case 7:
                for(int i = 0; i < lista.Count; i++)
                {
                    Console.WriteLine("{0} / {1}", lista[i].Nome, lista[i].Link);
                }
                break;
            default:
                break;
        }
    } while (escolha != 8);
  }
 }
}
```

Questão 2 Celula - Celula

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão2PorCelula
{
    internal class Celula
    {
        private Site valor;

        public Site Valor
        {
            get { return valor; }
            set { valor = value; }
        }
```

```
        private Celula proximo;

        public Celula Proximo
        {
            get { return proximo; }
            set { proximo = value; }
        }
        public Celula()
        {
            valor = new Site();
            proximo = null;
        }
        public Celula(Site valor)
        {
            this.valor = valor;
        }
    }
}
```

Questão 2 Celula - Inteiro

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão2PorCelula
{
    internal class Site
    {
        private string nome;

        public string Nome
        {
            get { return nome; }
            set { nome = value; }
        }
        private string link;

        public string Link
        {
            get { return link; }
```

```
            set { link = value; }
        }

        public Site()
        {
            nome = null;
            link = null;
        }
        public Site(string nome, string link)
        {
            this.nome = nome;
            this.link = link;
        }
        public void Imprimir()
        {
            Console.WriteLine("Site: {0}:{1}", nome, link);
        }
    }
}
```

Questão 2 Celula - Lista

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão2PorCelula
{
    internal class Lista
    {
        private Celula primeiro, ultimo;
        private int tamanho;
        public int Tamanho
        {
            get { return tamanho; }
        }
        public Lista()
        {
            Celula sentinela = new Celula();
            primeiro = sentinela;
            ultimo = sentinela;
```

```
    tamanho = 0;
}
public bool Vazia()
{
    if (primeiro == ultimo)
    {
        return true;
    }
    else
    {
        return false;
    }
}
public void Inserir(int pos, Site valor)
{
    if (pos >= 0 && pos <= tamanho)
    {
        Celula novaCelula = new Celula(valor);
        Celula anterior = primeiro;
        Celula proximaCelula;
        for (int i = 0; i < pos; i++)
        {
            anterior = anterior.Proximo;
        }
        proximaCelula = anterior.Proximo;
        anterior.Proximo = novaCelula;
        novaCelula.Proximo = proximaCelula;
        if (pos == tamanho)
        {
            ultimo = novaCelula;
        }
        tamanho++;
    }
    else
    {
        throw new Exception("Posição inválida");
    }
}
public Site Remover(int pos)
{
    if (!Vazia())
    {
        if (pos >= 0 && pos < tamanho)
        {
```

```
            Celula anterior = primeiro;
            for (int i = 0; i < pos; i++)
            {
                anterior = anterior.Proximo;
            }
            Celula celulaRemovida = anterior.Proximo;
            Celula proximaCelula = celulaRemovida.Proximo;
            anterior.Proximo = proximaCelula;
            celulaRemovida.Proximo = null;
            if (celulaRemovida == ultimo)
            {
                ultimo = anterior;
            }
            tamanho--;
            return celulaRemovida.Valor;
        }
        else
        {
            throw new Exception("Posição inválida");
        }
    }
    else
    {
        throw new Exception("Vazia!");
    }
}
public Site RemoverDireto(Site horas)
{
    if (!Vazia())
    {
        Celula anterior = primeiro;
        for (int i = 0; anterior.Proximo.Valor != horas; i++)
        {
            anterior = anterior.Proximo;
        }
        Celula removida = anterior.Proximo;
        Celula proxima = removida.Proximo;
        removida.Proximo = null;
        if (ultimo == removida)
        {
            ultimo = anterior;
        }
        tamanho--;
        return removida.Valor;
```

```
        }
        else
        {
            throw new Exception("Fila Vazia!");
        }
    }
    public int Contar(Site horas)
    {
        if (!Vazia())
        {
            Celula aux = primeiro;
            int cont = 0;
            while (aux != null)
            {
                if (aux.Valor == horas)
                {
                    cont++;
                }
            }
            return cont;
        }
        else
        {
            throw new Exception("Lista Vazia!");
        }

    }
    public void Imprimir()
    {
        if (!Vazia())
        {
            Celula aux = primeiro.Proximo;
            while (aux != null)
            {
                aux.Valor.Imprimir();
                aux = aux.Proximo;
            }
        }
        else
        {
            throw new Exception("Lista vazia!");
        }
    }
}
```

}

Questão 2 Celula - Main

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Security.Policy;
using System.Text;
using System.Threading.Tasks;

namespace Questão2PorCelula
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Lista lista = new Lista();
            int escolha, pos;
            string nome, link, removido;
            Site site;
            do
            {
                Console.WriteLine("1) Inserir um Site no início da lista\r\n2) Inserir um Site no final da lista\r\n3) Inserir um Site numa posição específica da lista\r\n4) Remover o primeiro Site da lista (Imprimir o nome do site removido)\r\n5) Remover o último Site da lista (Imprimir o nome do site removido)\r\n6) Remover um Site de uma posição específica da lista (Imprimir o nome do site removido)\r\n7) Mostrar o nome e o link de todos os sites da lista\r\n8) Encerrar o programa");
                escolha = int.Parse(Console.ReadLine());
                switch (escolha)
                {
                    case 1:
                        Console.WriteLine("Insira o nome e o link do site");
                        nome = Console.ReadLine();
                        link = Console.ReadLine();
                        site = new Site(nome, link);
                        lista.Inserir(0,site);
                        break;
                    case 2:
                        Console.WriteLine("Insira o nome e o link do site");
                        nome = Console.ReadLine();
```

```
                    link = Console.ReadLine();
                    site = new Site(nome, link);
                    lista.Inserir(lista.Tamanho,site);
                    break;
                case 3:
                    Console.WriteLine("Insira o nome, o link do site e a posição que
deseja inserir");
                    nome = Console.ReadLine();
                    link = Console.ReadLine();
                    pos = int.Parse(Console.ReadLine());
                    site = new Site(nome, link);
                    lista.Inserir(pos, site);
                    break;
                case 4:
                    removido = lista.Remover(0).Nome;
                    Console.WriteLine("Item {0} removido com sucesso", removido);
                    break;
                case 5:
                    removido = removido = lista.Remover(lista.Tamanho - 1).Nome;
                    Console.WriteLine("Item {0} removido com sucesso", removido);
                    break;
                case 6:
                    Console.WriteLine("Insira a posição que deseja remover: ");
                    pos = int.Parse(Console.ReadLine());
                    removido = lista.Remover(pos).Nome;
                    Console.WriteLine("Item {0} removido com sucesso", removido);
                    break;
                case 7:
                    lista.Imprimir();
                    break;
                default:
                    break;
            }
        } while (escolha != 8);
    }
  }
}
```

Questão 2 Vetor - Inteiro

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace Questão2PorVetor
{
    internal class Site
    {
        private string nome;

        public string Nome
        {
            get { return nome; }
            set { nome = value; }
        }
        private string link;

        public string Link
        {
            get { return link; }
            set { link = value; }
        }

        public Site()
        {
            nome = null;
            link = null;
        }
        public Site(string nome, string link)
        {
            this.nome = nome;
            this.link = link;
        }
        public void Imprimir()
        {
            Console.WriteLine("Site: {0}:{1}", nome, link);
        }
    }
}
```

Questão 2 Vetor - Vetor

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão2PorVetor
{
    internal class ListaVetor
    {
        private int primeiro;
        private int ultimo;
        private int tamanho;
        private Site[] lista;
        public ListaVetor(int size)
        {
            primeiro = 0;
            ultimo = 0;
            tamanho = 0;
            lista = new Site[size];
        }
        public int Tamanho
        {
            get { return tamanho; }
        }
        public bool Cheia()
        {
            if(ultimo == lista.Length)
            {
                return true;
            }
            else
            {
                return false;
            }
        }
        public bool Vazia()
        {
            if(primeiro == ultimo)
            {
                return true;
            }
            else
            {
                return false;
            }
```

```
}
public void Inserir(int pos, Site valor)
{
    if (!Cheia())
    {
        if(pos >= 0 && pos <= tamanho)
        {
            for(int i = ultimo; i > pos; i--)
            {
                lista[i] = lista[i - 1];
            }
            lista[pos] = valor;
            tamanho++;
            ultimo++;
        }
    }
    else
    {
        throw new Exception("Lista cheia");
    }
}
public Site Remover(int pos)
{
    if (!Vazia())
    {
        if(pos >= 0 && pos < tamanho)
        {
            Site removido = lista[pos];
            tamanho--;
            for(int i = pos; i < tamanho; i++)
            {
                lista[i] = lista[i + 1];
            }
            ultimo--;
            return removido;
        }
        else
        {
            throw new Exception("Fila Vazia!");
        }
    }
    else
    {
        throw new Exception("Fila Vazia!");
```

```
        }
    }
    public void Imprimir()
    {
        for(int i = primeiro; i < ultimo; i++)
        {
            Console.WriteLine("{0} / {0}", lista[i].Nome, lista[i].Link);
        }
    }
  }
}
```

Questão 2 Vetor - Main

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão2PorVetor
{
  internal class Program
  {
    static void Main(string[] args)
    {
      ListaVetor lista = new ListaVetor(20);
      int escolha, pos;
      string nome, link, removido;
      Site site;
      do
      {
        Console.WriteLine("1) Inserir um Site no início da lista\r\n2) Inserir um Site no final da lista\r\n3) Inserir um Site numa posição específica da lista\r\n4) Remover o primeiro Site da lista (Imprimir o nome do site removido)\r\n5) Remover o último Site da lista (Imprimir o nome do site removido)\r\n6) Remover um Site de uma posição específica da lista (Imprimir o nome do site removido)\r\n7) Mostrar o nome e o link de todos os sites da lista\r\n8) Encerrar o programa");
        escolha = int.Parse(Console.ReadLine());
        switch (escolha)
        {
```

```
                case 1:
                    Console.WriteLine("Insira o nome e o link do site");
                    nome = Console.ReadLine();
                    link = Console.ReadLine();
                    site = new Site(nome, link);
                    lista.Inserir(0, site);
                    break;
                case 2:
                    Console.WriteLine("Insira o nome e o link do site");
                    nome = Console.ReadLine();
                    link = Console.ReadLine();
                    site = new Site(nome, link);
                    lista.Inserir(lista.Tamanho, site);
                    break;
                case 3:
                    Console.WriteLine("Insira o nome, o link do site e a posição que
deseja inserir");
                    nome = Console.ReadLine();
                    link = Console.ReadLine();
                    pos = int.Parse(Console.ReadLine());
                    site = new Site(nome, link);
                    lista.Inserir(pos, site);
                    break;
                case 4:
                    removido = lista.Remover(0).Nome;
                    Console.WriteLine("Item {0} removido com sucesso", removido);
                    break;
                case 5:
                    removido = removido = lista.Remover(lista.Tamanho - 1).Nome;
                    Console.WriteLine("Item {0} removido com sucesso", removido);
                    break;
                case 6:
                    Console.WriteLine("Insira a posição que deseja remover: ");
                    pos = int.Parse(Console.ReadLine());
                    removido = lista.Remover(pos).Nome;
                    Console.WriteLine("Item {0} removido com sucesso", removido);
                    break;
                case 7:
                    lista.Imprimir();
                    break;
                default:
                    break;
            }
        } while (escolha != 8);
```

```
        }
    }
}
```

Questão 3 Celula - Celula

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão3Lista11
{
    internal class CelulaDupla
    {
        private Inteiro valor;

        public Inteiro Valor
        {
            get { return valor; }
            set { valor = value; }
        }
        private CelulaDupla proximo;

        public CelulaDupla Proximo
        {
            get { return proximo; }
            set { proximo = value; }
        }
        private CelulaDupla anterior;

        public CelulaDupla Anterior
        {
            get { return anterior; }
            set { anterior = value; }
        }
        public CelulaDupla()
        {
            valor = new Inteiro();
            proximo = null;
            anterior = null;
        }
```

```
        public CelulaDupla(Inteiro valor)
        {
            this.valor = valor;
            proximo = null;
            anterior = null;
        }
    }
}
```

Questão 3 Celula - Inteiro

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão3Lista11
{
    internal class Inteiro
    {
        private string item;

        public string Item
        {
            get { return item; }
            set { item = value; }
        }
        public Inteiro()
        {
            item = null;
        }
        public Inteiro(string item)
        {
            this.item = item;
        }
        public void Imprimir()
        {
            Console.WriteLine("Item: {0}", item);
        }
    }
}
```

Questão 3 Celula - Lista

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão3Lista11
{
    internal class Lista
    {
        private CelulaDupla primeiro;
        private CelulaDupla ultimo;
        private int tamanho;
        public Lista()
        {
            CelulaDupla sentinela = new CelulaDupla();
            primeiro = sentinela;
            ultimo = sentinela;
            tamanho = 0;
        }
        public void Inserir(int pos, Inteiro valor)
        {
            if (pos >= 0 && pos <= tamanho)
            {
                CelulaDupla anterior = primeiro;
                for (int i = 0; i < pos; i++)
                {
                    anterior = anterior.Proximo;
                }
                CelulaDupla novaCelula = new CelulaDupla(valor);
                CelulaDupla proximaCelula = anterior.Proximo;
                anterior.Proximo = novaCelula;
                novaCelula.Proximo = proximaCelula;
                novaCelula.Anterior = anterior;
                if (pos == tamanho)
                {
                    ultimo = novaCelula;
                }
                tamanho++;
            }
            else
```

```
    {
        throw new Exception("Pos inválida");
    }
}
public bool Vazia()
{
    if (primeiro == ultimo)
    {
        return true;
    }
    else
    {
        return false;
    }
}
public Inteiro Remover(int pos)
{
    CelulaDupla anterior = primeiro;
    if (!Vazia())
    {
        if (pos >= 0 && pos < tamanho)
        {
            for (int i = 0; i < pos; i++)
            {
                anterior = anterior.Proximo;
            }
            CelulaDupla removida = anterior.Proximo;
            CelulaDupla proximaCelula = removida.Proximo;
            anterior.Proximo = proximaCelula;
            proximaCelula.Anterior = anterior;
            removida.Proximo = null;
            removida.Anterior = null;
            if (removida == ultimo)
            {
                ultimo = anterior;
            }
            tamanho--;
            return removida.Valor;
        }
        else
        {
            throw new Exception("Pos inválida");
        }
    }
```

```
    else
    {
        throw new Exception("Lista Vazia");
    }
}
public void InserirFinal(Inteiro valor)
{
    CelulaDupla novaCelula = new CelulaDupla(valor);
    ultimo.Proximo = novaCelula;
    novaCelula.Anterior = ultimo;
    ultimo = novaCelula;
    tamanho++;
}
public Inteiro RemoverFinal()
{
    if (!Vazia())
    {
        CelulaDupla penultima = ultimo.Anterior;
        CelulaDupla removida = ultimo;
        penultima.Proximo = null;
        removida.Anterior = null;
        ultimo = penultima;
        tamanho--;
        return removida.Valor;
    }
    else
    {
        throw new Exception("Lista Vazia");
    }
}
public void Imprimir()
{
    if (!Vazia())
    {
        CelulaDupla aux = primeiro.Proximo;
        while (aux != null)
        {
            aux.Valor.Imprimir();
            aux = aux.Proximo;
        }
    }
    else
    {
        throw new Exception("Lista vazia!");
```

```
    }
}
public void ImprimirInverso()
{
    if (!Vazia())
    {
        CelulaDupla aux = ultimo;
        while (aux != null)
        {
            aux.Valor.Imprimir();
            aux = aux.Anterior;
        }
    }
    else
    {
        throw new Exception("Lista vazia!");
    }
}
public int Position(Inteiro nome)
{
    if (!Vazia())
    {
        CelulaDupla aux = primeiro.Proximo;
        int cont = 0;
        bool resp = false;
        while (aux != null && resp == false)
        {
            aux = aux.Proximo;
            cont++;
            if (aux.Valor.Item == nome.Item)
            {
                resp = true;
            }
        }
        if(cont == 0) {
            throw new Exception("Inválido");
        }
        else
        {
            return cont + 1; // O loop para uma música antes, por isso somei 1
        }
    }
    else
    {
```

```
            throw new Exception("Lista vazia!");
        }
    }
    public Inteiro Anterior(Inteiro nome)
    {
        if (!Vazia())
        {
            CelulaDupla aux = primeiro;
            while (aux.Proximo.Valor.Item != nome.Item && aux != null)
            {
                aux = aux.Proximo;
            }
            return aux.Valor;
        }
        else
        {
            throw new Exception("Lista vazia!");
        }
    }
    public Inteiro Posterior(Inteiro nome)
    {
        if (!Vazia())
        {
            CelulaDupla aux = primeiro;
            while (aux.Proximo.Valor.Item != nome.Item && aux != null)
            {
                aux = aux.Proximo;
            }
            CelulaDupla desejada = aux.Proximo;
            return desejada.Proximo.Valor;
        }
        else
        {
            throw new Exception("Lista vazia!");
        }
    }
  }
}
```

Questão 3 Celula - Main

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Questão3Lista11
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Lista lista = new Lista();
            Inteiro musica;
            string removido, nome;
            int escolha, pos;
            do
            {
                Console.WriteLine("1. Inserir uma música no final da lista\r\n2. Inserir uma música no início da lista\r\n3. Inserir uma música numa posição específica da lista\r\n4. Remover a música do início da lista\r\n5. Remover a música do final da lista\r\n6. Remover uma música de uma posição específica da lista\r\n7. Listar todas as músicas da lista\r\n8. Listar todas as músicas da lista na ordem inversa (O programa deve imprimir da última música na lista até\r\na primeira)\r\n9. Pesquisar uma música na lista (O usuário deve informar uma música. O programa deve imprimir a\r\nposição da música informada).\r\n10. Pesquisar música anterior (O usuário deve informar uma música. O programa deve imprimir a música\r\nanterior a música informada)\r\n11. Pesquisar música posterior (O usuário deve informar uma música. O programa deve imprimir a música\r\nposterior a música informada)\r\n12. Encerrar o programa");
                escolha = int.Parse(Console.ReadLine());
                switch (escolha)
                {
                    case 1:
                        Console.WriteLine("Insira o nome da música");
                        nome = Console.ReadLine();
                        musica = new Inteiro(nome);
                        lista.InserirFinal(musica);
                        break;
                    case 2:
                        Console.WriteLine("Insira o nome da música");
                        nome = Console.ReadLine();
                        musica = new Inteiro(nome);
                        lista.Inserir(0, musica);
                        break;
                    case 3:
```

```
            Console.WriteLine("Insira o nome da música e a posição desejada");
            nome = Console.ReadLine();
            musica = new Inteiro(nome);
            pos = int.Parse(Console.ReadLine());
            lista.Inserir(pos, musica);
            break;
        case 4:
            removido = lista.Remover(0).Item;
            Console.WriteLine("{0} removida com sucesso", removido);
            break;
        case 5:
            removido = lista.RemoverFinal().Item;
            Console.WriteLine("{0} removida com sucesso", removido);
            break;
        case 6:
            Console.WriteLine("Insira a posição desejada");
            pos = int.Parse(Console.ReadLine());
            removido = lista.Remover(pos).Item;
            Console.WriteLine("{0} removida com sucesso", removido);
            break;
        case 7:
            lista.Imprimir();
            break;
        case 8:
            lista.ImprimirInverso();
            break;
        case 9:
            Console.WriteLine("Insira a música desejada");
            nome = Console.ReadLine();
            musica = new Inteiro(nome);
            Console.WriteLine("Posição: "+lista.Position(musica));
            break;
        case 10:
            Console.WriteLine("Insira a música desejada");
            nome = Console.ReadLine();
            musica = new Inteiro(nome);
            Console.WriteLine("Anterior: " + lista.Anterior(musica).Item);
            break;
        case 11:
            Console.WriteLine("Insira a música desejada");
            nome = Console.ReadLine();
            musica = new Inteiro(nome);
            Console.WriteLine("Posterior: " + lista.Posterior(musica).Item);
            break;
```

```
                default:
                    break;
            }
        } while (escolha != 12);
    }
  }
}
```