João Pedro Anicio Botelho

Roteiro

1. Crie o ArrayList AL1 sem definir a capacidade inicial.
```
ArrayList AL1 = new ArrayList();
```

2. Imprima a capacidade e a quantidade de elementos.
```
Console.WriteLine("Capacidade: {0}, Quantidade: {1}", AL1.Capacity, AL1.Count);
```

3. Adicione os números 19, 7 e 11.
```
AL1.Add(19);
AL1.Add(7);
AL1.Add(11);
```

4. Imprima a capacidade e a quantidade de elementos.
```
Console.WriteLine("Capacidade: {0}, Quantidade: {1}", AL1.Capacity, AL1.Count);
```

5. Adicione os números 5, 7 e 17.
```
AL1.Add(5);
AL1.Add(7);
AL1.Add(17);
```

6. Imprima a capacidade e a quantidade de elementos.
```
Console.WriteLine("Capacidade: {0}, Quantidade: {1}", AL1.Capacity, AL1.Count);
```

7. Imprima os elementos do ArrayList usando o comando foreach.
```
Console.Write("[ ");
foreach (int n in list) {
  Console.Write(n + " ");
}
Console.WriteLine("]");
```

8. Adicione o número 5 nas posições 0, 2 e 10. Ocorreu algum problema? Se sim, anote qual foi o motivo e resolva esse problema.
```
AL1.Insert(0, 5);
AL1.Insert(2, 5);
AL1.Add(5); // Usando Add porque a posição 10 não existe
```

   Para inserir na posição deve ser usado Insert e não o Add, e deve ser inserido em uma posição que seja menor ou igual ao último item

9. Imprima os elementos do ArrayList usando o comando for.
```
Console.Write("[ ");
for (int i=0; i < list.Count; i++) {
  Console.Write(list[i] + " ");
}
Console.WriteLine("]");
```

10. Imprima a primeira e a última posição que contém o número 7.
```
Console.WriteLine("Primeira posição com número sete: {0}, última posição: {1}",
AL1.IndexOf(7), AL1.LastIndexOf(7));
```

11. Imprima todas as posições que contém o número 5.
```
Console.Write("Todas posições com número cinco: ");
for (int i=0; i < AL1.Count; i++) {
  if (Convert.ToInt32(AL1[i]) == 5)
    Console.Write(i + " ");
}
Console.Write("\n");
```

12. Adicione os números 5, 23, 47, 5 e 5.

```
AL1.Add(5);
AL1.Add(23);
AL1.Add(47);
AL1.Add(5);
AL1.Add(5);
```

13. Imprima os elementos do ArrayList usando o comando while.

```
int cont = 0;
Console.Write("[ ");

while (cont < list.Count) {
  Console.Write(list[cont] + " ");
  cont++;
}

Console.WriteLine("]");
```

14. Imprima as posições do número 5 retornada pelos métodos: BinarySeach(), IndexOf() e LastIndexOf()

```
Console.WriteLine("BinarySeach: {0}, IndexOf: {1}, LastIndexOf: {2}",
AL1.BinarySearch(5), AL1.IndexOf(5), AL1.LastIndexOf(5));
```

15. Ordene os elementos do ArrayList.

```
AL1.Sort();
```

16. Imprima os elementos do ArrayList (use o comando que você desejar).

```
Console.Write("[ ");
foreach (int n in list) {
  Console.Write(n + " ");
}
Console.WriteLine("]");
```

17. Remova o número 23.

```
AL1.Remove(23);
```

18. Imprima os elementos do ArrayList (use o comando que você desejar).

```
Console.Write("[ ");
for (int i=0; i < list.Count; i++) {
  Console.Write(list[i] + " ");
}
Console.WriteLine("]");
```

19. Remova o elemento da posição 7.

```
AL1.RemoveAt(7);
```

20. Imprima os elementos do ArrayList (use o comando que você desejar).

```
int cont = 0;
Console.Write("[ ");

while (cont < list.Count) {
  Console.Write(list[cont] + " ");
  cont++;
}

Console.WriteLine("]");
```

21. Remova os elementos das posições 2, 3 e 4.

```
AL1.RemoveAt(4);
AL1.RemoveAt(3);
AL1.RemoveAt(2);
```

22. Imprima os elementos do ArrayList (use o comando que você desejar).

```
Console.Write("[ ");
foreach (int n in list) {
  Console.Write(n + " ");
}
Console.WriteLine("]");
```

23. Inverta os elementos do ArrayList.

```
AL1.Reverse();
```

24. Imprima os elementos do ArrayList (use o comando que você desejar).

```
Console.Write("[ ");
for (int i=0; i < list.Count; i++) {
  Console.Write(list[i] + " ");
}
Console.WriteLine("]");
```

25. Remova todos os elementos do ArrayList.

```
AL1.Clear();
```

26. Imprima os elementos do ArrayList (use o comando que você desejar).

```
int cont = 0;
Console.Write("[ ");

while (cont < list.Count) {
  Console.Write(list[cont] + " ");
  cont++;
}

Console.WriteLine("]");
```

27. Imprima a capacidade e a quantidade de elementos.

```
Console.WriteLine("Capacidade: {0}, Quantidade: {1}", AL1.Capacity, AL1.Count);
```

Lista de exercícios

Coleções .NET

Atenção, cada uma das questões deve ter 3 soluções, uma para ***ArrayList***, outra para ***Queue*** e outra para ***Stack***. Para todos os exercícios, considere soluções não-destrutivas, ou seja, os dados devem ser mantidos na ordem original.

1 - Crie um ArrayList e adicione 10 valores inteiros digitados pelo usuário. Ao final, imprimatodos os elementos.

ArrayList

```
ArrayList al = new ArrayList();
for (int i=0; i<10; i++) {
  Console.WriteLine("Informe o {0}° valor inteiro: ", i+1);
  al.Add(Convert.ToInt32(Console.ReadLine()));
}

Console.Write("\nArrayList: [");
for (int i=0; i<al.Count; i++) {
  Console.Write("{0}", i==0 ? al[i] : ", " + al[i]);
}
Console.WriteLine("]\n");
```

Queue

```
Queue qu = new Queue();
for (int i=0; i<10; i++) {
  Console.WriteLine("Informe o {0}° valor inteiro: ", i+1);
  qu.Enqueue(Convert.ToInt32(Console.ReadLine()));
}

int cont = 0;
Console.Write("\nQueue: [");
foreach (object obj in qu) {
  Console.Write("{0}", cont==0 ? obj : ", " + obj);
  cont++;
}
Console.WriteLine("]\n");
```

Stack

```
Stack st = new Stack();
for (int i=0; i<10; i++) {
  Console.WriteLine("Informe o {0}° valor inteiro: ", i+1);
  st.Push(Convert.ToInt32(Console.ReadLine()));
}

int cont = 0;
Console.Write("\nStack: [");
foreach (object obj in st) {
  Console.Write("{0}", cont==0 ? obj : ", " + obj);
  cont++;
}
Console.WriteLine("]\n");
```

2 - Crie um ArrayList e adicione 10 strings digitados pelo usuário. Ao final, imprima todos oselementos.

ArrayList

```
ArrayList al = new ArrayList();
for (int i=0; i<10; i++) {
  Console.WriteLine("Informe a {0}° string: ", i+1);
  al.Add(Console.ReadLine());
}

Console.Write("\nArrayList: [\n");
for (int i=0; i<al.Count; i++) {
  Console.Write(" {0}\n", i==al.Count-1 ? al[i] : al[i] + ",");
}
Console.WriteLine("]\n");
```

Queue

```
Queue qu = new Queue();
for (int i=0; i<10; i++) {
  Console.WriteLine("Informe a {0}° string: ", i+1);
  qu.Enqueue(Console.ReadLine());
}

int cont = 0;
Console.Write("\nQueue: [\n");
foreach (object obj in qu) {
  Console.Write(" {0}\n", cont==qu.Count-1 ? obj : obj + ",");
  cont++;
}
Console.WriteLine("]\n");
```

Stack

```
Stack st = new Stack();
for (int i=0; i<10; i++) {
  Console.WriteLine("Informe a {0}° string: ", i+1);
  st.Push(Console.ReadLine());
}

int cont = 0;
Console.Write("\nStack: [\n");
foreach (object obj in st) {
  Console.Write(" {0}\n", cont==st.Count-1 ? obj : obj + ",");
  cont++;
}
Console.WriteLine("]\n");
```

3 - Crie um ArrayList e adicione os números ímpares no intervalo entre 1 a 100. Calcule asoma dos números usando o comando **for**.

ArrayList

```
ArrayList al = new ArrayList();
for(int i=1; i<=100; i+=2) {
  al.Add(i);
}

int soma = 0;
for(int i=0; i<al.Count; i++) {
  soma += Convert.ToInt32(al[i]);
}

Console.WriteLine("\nSoma Array: {0}\n", soma);
```

Queue

```
Queue qu = new Queue();
for(int i=1; i<=100; i+=2) {
  qu.Enqueue(i);
}

int soma = 0;
int length = qu.Count;
for(int i=0; i<length; i++) {
  soma += Convert.ToInt32(qu.Dequeue());
}

Console.WriteLine("\nSoma Queue: {0}\n", soma);
```

Stack

```
Stack st = new Stack();
for(int i=1; i<=100; i+=2) {
  st.Push(i);
}

int soma = 0;
int length = st.Count;
for(int i=0; i<length; i++) {
  soma += Convert.ToInt32(st.Pop());
}

Console.WriteLine("\nSoma Stack: {0}\n", soma);
```

4 - Crie um ArrayList e adicione os números pares no intervalo entre 1 a 100. Calcule a soma dos números usando o comando **while**.

ArrayList

```
ArrayList al = new ArrayList();
for(int i=2; i<=100; i+=2) {
  al.Add(i);
}

int soma=0, cont=0;
while (cont < al.Count) {
  soma += Convert.ToInt32(al[cont]);
  cont++;
}

Console.WriteLine("\nSoma Array: {0}\n", soma);
```

Queue

```
Queue qu = new Queue();
for(int i=2; i<=100; i+=2) {
  qu.Enqueue(i);
}

int soma=0, cont=0;
int length = qu.Count;
while (cont < length) {
  soma += Convert.ToInt32(qu.Dequeue());
  cont++;
}

Console.WriteLine("\nSoma Queue: {0}\n", soma);
```

Stack

```
Stack st = new Stack();
for(int i=2; i<=100; i+=2) {
  st.Push(i);
}

int soma=0, cont=0;
int length = st.Count;
while (cont < length) {
  soma += Convert.ToInt32(st.Pop());
  cont++;
}

Console.WriteLine("\nSoma Stack: {0}\n", soma);
```

5 - Crie um ArrayList e adicione os números no intervalo entre 1 a 100. Calcule a soma dos números usando o comando **do while**.

ArrayList

```
ArrayList al = new ArrayList();
for(int i=1; i<=100; i++) {
  al.Add(i);
}

int soma=0, cont=0;
do {
  soma += Convert.ToInt32(al[cont]);
  cont++;
} while (cont < al.Count);

Console.WriteLine("\nSoma Array: {0}\n", soma);
```

Queue

```
Queue qu = new Queue();
for(int i=1; i<=100; i++) {
  qu.Enqueue(i);
}

int soma=0, cont=0;
int length = qu.Count;
do {
  soma += Convert.ToInt32(qu.Dequeue());
  cont++;
} while (cont < length);

Console.WriteLine("\nSoma Queue: {0}\n", soma);
```

Stack

```
Stack st = new Stack();
for(int i=1; i<=100; i++) {
  st.Push(i);
}

int soma=0, cont=0;
int length = st.Count;
do {
  soma += Convert.ToInt32(st.Pop());
  cont++;
} while (cont < length);

Console.WriteLine("\nSoma Stack: {0}\n", soma);
```

6 - Crie um ArrayList e adicione os números pares no intervalo entre 1 a 100. Calcule a soma dos números usando o comando **foreach**.

ArrayList

```
ArrayList al = new ArrayList();
for(int i=2; i<=100; i+=2) {
  al.Add(i);
}

int soma=0;
foreach (int item in al) {
  soma += item;
}

Console.WriteLine("\nSoma Array: {0}\n", soma);
```

Queue

```
Queue qu = new Queue();
for(int i=2; i<=100; i+=2) {
  qu.Enqueue(i);
}

int soma=0;
foreach (object item in qu) {
  soma += Convert.ToInt32(item);
}

Console.WriteLine("\nSoma Queue: {0}\n", soma);
```

Stack

```
Stack st = new Stack();
for(int i=2; i<=100; i+=2) {
  st.Push(i);
}

int soma=0;
foreach (object item in st) {
  soma += Convert.ToInt32(item);
}

Console.WriteLine("\nSoma Stack: {0}\n", soma);
```

7 - Crie um ArrayList contendo os seguintes elementos (5, 13, 19, 31, 3, 7, 11, 5, 57, 13, 5).Faça uma função que apague TODAS as ocorrências de um determinado elemento. Use essa função para apagar todas as ocorrências do número 5 e 13.

Dica: você pode criar um ArrayList e já inicializá-lo com alguns elementos conforme o exemplo abaixo:

```
ArrayList AL = new ArrayList() {1, 2, 3, 4, 5};
```

ArrayList

```
    ArrayList al = new ArrayList() {5, 13, 19, 31, 3, 7, 11, 5, 57, 13, 5};
    removeItemsArray(al, 5);
    removeItemsArray(al, 13);

    Print(al);
```

```
  private static void removeItemsArray(ArrayList al, int item) {
    int length = al.Count;
    for (int i=0; i<length; i++) {
      if (Convert.ToInt32(al[i]) == item) {
        al.RemoveAt(i);
        length--;
      }
    }
  }
```

Queue

```
    Queue qu = new Queue();
    qu.Enqueue(5);
    qu.Enqueue(13);
    qu.Enqueue(19);
    qu.Enqueue(31);
    qu.Enqueue(3);
    qu.Enqueue(7);
    qu.Enqueue(11);
    qu.Enqueue(5);
    qu.Enqueue(57);
    qu.Enqueue(13);
    qu.Enqueue(5);

    removeItemsQueue(qu, 5);
    removeItemsQueue(qu, 13);

    Print(qu);
```

```
  private static void removeItemsQueue(Queue tad, int item) {
    object?[] aux = tad.ToArray();
    tad.Clear();
    for(int i=aux.Length-1; i>=0; i--) {
      if (Convert.ToInt32(aux[i]) != item)
        tad.Enqueue(aux[i]);
    }
  }
```

Stack

```
    Stack st = new Stack();
    st.Push(5);
    st.Push(13);
    st.Push(19);
    st.Push(31);
    st.Push(3);
    st.Push(7);
    st.Push(11);
    st.Push(5);
    st.Push(57);
    st.Push(13);
    st.Push(5);

    removeItemsStack(st, 5);
    removeItemsStack(st, 13);

    Print(st);
```

```
    object?[] aux = tad.ToArray();
    tad.Clear();
    for(int i=aux.Length-1; i>=0; i--) {
      if (Convert.ToInt32(aux[i]) != item)
        tad.Push(aux[i]);
    }
```

Função Print

```
  private static void Print(IEnumerable tad) {
    Console.Write("[");

    foreach(var item in tad) {
      Console.Write(" {0}", item);
    }

    Console.WriteLine(" ]\n");
  }
```

8 – Faça um programa que **leia n números** inteiros e os armazene em um ArrayList.Calcule a soma e a média aritmética (use o comando FOR e depois o FOREACH).

```
  private static int selectForOrForeach() {
    Console.WriteLine("\n|| Selecione uma opção ||");
    Console.WriteLine("|| = = = = = = = = = = ||");
    Console.WriteLine("|| 1 - Usar For        ||");
    Console.WriteLine("|| 2 - Usar ForEach    ||");
    Console.WriteLine("|| = = = = = = = = = = ||");

    int resp = Convert.ToInt32(Console.ReadLine());

    if(resp != 1 && resp != 2) {
      Console.WriteLine("Opção Inválida.");
      selectForOrForeach();
    }

    return resp;
  }
```

```
      Console.WriteLine("Quantos números serão inseridos:");
      n = Convert.ToInt32(Console.ReadLine());

      ArrayList al = new ArrayList();

      for(int i=0; i<n; i++) {
        Console.WriteLine("Insira o {0}° número inteiro:", i+1);
        al.Add(Convert.ToInt32(Console.ReadLine()));
      }

      int method = selectForOrForeach();
      int soma = 0;

      if (method == 1) {
        for(int i=0; i<al.Count; i++) {
          soma += Convert.ToInt32(al[i]);
        }
      } else {
        foreach(int num in al) {
          soma += num;
        }
      }
      Console.WriteLine((method == 1 ? "FOR" : "FOREACH") + " - Soma: {0},
Média: {1}\n", soma, soma/al.Count);
```

9 – Faça um programa que **leia n números** inteiros e os armazene em um Queue. Calculea soma e a média aritmética (use o comando FOREACH para iterar sobre os elementos).

```
      Console.WriteLine("Quantos números serão inseridos:");
      n = Convert.ToInt32(Console.ReadLine());

      Queue qu = new Queue();

      for(int i=0; i<n; i++) {
        Console.WriteLine("Insira o {0}° número inteiro:", i+1);
        qu.Enqueue(Convert.ToInt32(Console.ReadLine()));
      }

      int soma = 0;
      foreach(object obj in qu) {
        soma += Convert.ToInt32(obj);
      }

      Console.WriteLine("Soma: {0}, Média: {1}\n", soma, soma/qu.Count);
```

10 – Faça um programa que **leia n números** inteiros e os armazene em um Stack. Calculea soma e a média aritmética (use o comando FOREACH para iterar sobre os elementos).

```
      Stack st = new Stack();

      for(int i=0; i<n; i++) {
        Console.WriteLine("Insira o {0}° número inteiro:", i+1);
        st.Push(Convert.ToInt32(Console.ReadLine()));
      }

      int soma = 0;
      foreach(object obj in st) {
        soma += Convert.ToInt32(obj);
      }

      Console.WriteLine("Soma: {0}, Média: {1}\n", soma, soma/st.Count);
```

11 – Faça um programa que preencha um ArrayList com elementos de diferentes tipos (int, double, float, boolean, String). Tente calcular a soma dos elementos. Evidentemente, isso irá provocar uma mensagem de erro. Que mensagem o Visual Studio retorna?

Dica: você pode criar um ArrayList e já inicializá-lo com alguns elementos conforme o exemplo abaixo:

```
ArrayList AL = new ArrayList() { 1, 2, "AED", new Queue(), "teste", 3.14 };
```

ArrayList

```
      ArrayList al = new ArrayList() { 1, 2.5, 10.10, true, "João" };

      var soma = 0;
      foreach(var item in al) {
        soma += Convert.ToInt32(item);
      }
      Console.WriteLine("Soma: {0}", soma);
      // ERRO: Unhandled exception. System.FormatException: Input string was not
in a correct format.
```

Queue

```
      Queue qu = new Queue();
      qu.Enqueue(1);
      qu.Enqueue(1.5);
      qu.Enqueue(10.10);
      qu.Enqueue(true);
      qu.Enqueue("João");

      var soma = 0;
      foreach(var item in qu) {
        soma += Convert.ToInt32(item);
      }
      Console.WriteLine("Soma: {0}", soma);
      // ERRO: Unhandled exception. System.FormatException: Input string was not
in a correct format.
```

Stack

```
      Stack st = new Stack();
      st.Push(1);
      st.Push(1.5);
      st.Push(10.10);
      st.Push(true);
      st.Push("João");

      var soma = 0;
      foreach(var item in st) {
        soma += Convert.ToInt32(item);
      }
      Console.WriteLine("Soma: {0}", soma);
      // ERRO: Unhandled exception. System.FormatException: Input string was not
in a correct format.
```

12 – Faça um programa que preencha um ArrayList com os números entre 1 e 25. Pede-se:

- Imprima todos os elementos
- Imprima todos os elementos em ordem invertida
- Imprima todos os elementos em posições ímpares (os elementos da posição 1, 3, 5, ...)
- Imprima todos os elementos ímpares
- Imprima apenas os elementos da primeira metade do vetor (posição 0 a 12).

OBS: você deve fazer esse programa 2 vezes. Primeiro usando o comando FOR e depois usando o comando FOREACH.

ArrayList

```
      ArrayList al = new ArrayList();

      for(int i=1; i<=25; i++) {
        al.Add(i);
      }

      int method = selectForOrForeach();
      int length = al.Count;

      Console.WriteLine("\nArray - {0}\n", method == 1 ? "FOR" : "FOREACH");

      Console.WriteLine("Imprime:");
      Print(al, method, length);

      al.Reverse();

      Console.WriteLine("Imprime Reverse:");
      Print(al, method, length);

      Console.WriteLine("Imprime Posições Ímpares:");
      PrintImpar(al, true, method, length);

      Console.WriteLine("Imprime Elementos Ímpares:");
      PrintImpar(al, false, method, length);

      Console.WriteLine("Imprime a Primeira Metade:");
      PrintHalf(al, length, method);
```

Queue

```
Queue qu = new Queue();

for(int i=1; i<=25; i++) {
  qu.Enqueue(i);
}

int method = selectForOrForeach();
int length = qu.Count;

Console.WriteLine("\nQueue - {0}\n", method == 1 ? "FOR" : "FOREACH");

Console.WriteLine("Imprime:");
Print(qu, method, length);

ReverseQueue(qu);

Console.WriteLine("Imprime Reverse:");
Print(qu, method, length);

Console.WriteLine("Imprime Posições Ímpares:");
PrintImpar(qu, true, method, length);

Console.WriteLine("Imprime Elementos Ímpares:");
PrintImpar(qu, false, method, length);

Console.WriteLine("Imprime a Primeira Metade:");
PrintHalf(qu, length, method);
```

Stack

```
Stack st = new Stack();

for(int i=1; i<=25; i++) {
  st.Push(i);
}

int method = selectForOrForeach();
int length = st.Count;

Console.WriteLine("\nStack - {0}\n", method == 1 ? "FOR" : "FOREACH");

Console.WriteLine("Imprime:");
Print(st, method, length);

ReverseStack(st);

Console.WriteLine("Imprime Reverse:");
Print(st, method, length);

Console.WriteLine("Imprime Posições Ímpares:");
PrintImpar(st, true, method, length);

Console.WriteLine("Imprime Elementos Ímpares:");
PrintImpar(st, false, method, length);

Console.WriteLine("Imprime a Primeira Metade:");
PrintHalf(st, length, method);
```

Funções Auxiliares

```
private static int selectForOrForeach() {
  Console.WriteLine("\n|| Selecione uma opção ||");
  Console.WriteLine("|| = = = = = = = = = = ||");
  Console.WriteLine("|| 1 - Usar For         ||");
  Console.WriteLine("|| 2 - Usar ForEach     ||");
  Console.WriteLine("|| = = = = = = = = = = ||");

  int resp = Convert.ToInt32(Console.ReadLine());

  if(resp != 1 && resp != 2) {
    Console.WriteLine("Opção Inválida.");
    selectForOrForeach();
  }

  return resp;
}
```

```
private static void Print(IEnumerable tad, int method, int length) {
  Console.Write("[");

  if (method == 1) {
    IEnumerator e = tad.GetEnumerator();
    for(int i=0; i<length; i++) {
      e.MoveNext();
      Console.Write(" {0}", e.Current);
    }
  } else {
    foreach(var item in tad) {
      Console.Write(" {0}", item);
    }
  }

  Console.WriteLine(" ]\n");
}
```

```
private static void PrintHalf(IEnumerable tad, int length, int method) {
  Console.Write("[");

  if (method == 1) {
    IEnumerator e = tad.GetEnumerator();
    for (int i=0; i<=length/2; i++) {
      e.MoveNext();
      Console.Write(" {0}", e.Current);
    }
  } else {
    int i = 0;
    foreach(var item in tad) {
      if (i > length/2) break;
      Console.Write(" {0}", item);
      i++;
    }
  }

  Console.WriteLine(" ]\n");
}
```

```
      private static void PrintImpar(IEnumerable tad, bool key, int method, int
length) {
      Console.Write("[");

      if (method == 1) {
        IEnumerator e = tad.GetEnumerator();
        for(int i=0; i<length; i++) {
          e.MoveNext();
          bool regra = (key ? i : Convert.ToInt32(e.Current)) % 2 != 0;
          if (regra)
            Console.Write(" {0}", e.Current);
        }
      } else {
        int i = 0;
        foreach(var item in tad) {
          bool regra = (key ? i : Convert.ToInt32(item)) % 2 != 0;
          if (regra)
            Console.Write(" {0}", item);
          i++;
        }
      }

      Console.WriteLine(" ]\n");
    }
```

```
    private static void ReverseQueue(Queue tad) {
      object?[] aux = tad.ToArray();
      tad.Clear();
      for(int i=aux.Length-1; i>=0; i--) {
        tad.Enqueue(aux[i]);
      }
    }
```

```
    private static void ReverseStack(Stack tad) {
      object?[] aux = tad.ToArray();
      tad.Clear();
      for(int i=aux.Length-1; i>=0; i--) {
        tad.Push(aux[i]);
      }
    }
```

13 - Faça um programa que gere uma coleção com **n** números inteiros aleatórios (o valor de **n** deve ser informado pelo usuário no início da execução do programa. Imprima os elementos da coleção.

Exemplo de geração de números aleatórios.

```
Random r = new Random();
int x = r.Next(); // Retorna um número aleatório positivo
int y = r.Next(100); // Retorna um número aleatório entre 0 e 99
int z = r.Next(50, 100); // Retorna um número aleatório entre 50 e 99
double w = r.NextDouble(); // Retorna um ponto-flutuante entre 0.0 e 1.0
```

ArrayList

```
    ArrayList al = new ArrayList();
    Random r = new Random();

    for(int i=0; i<n; i++) {
      int x = r.Next();
      al.Add(x);
    }

    Console.WriteLine("\nArrayList:");
    Print(al);
```

Queue

```
    Queue qu = new Queue();
    Random r = new Random();

    for(int i=0; i<n; i++) {
      int x = r.Next();
      qu.Enqueue(x);
    }

    Console.WriteLine("\nQueue:");
    Print(qu);
```

Stack

```
    Stack st = new Stack();
    Random r = new Random();

    for(int i=0; i<n; i++) {
      int x = r.Next();
      st.Push(x);
    }

    Console.WriteLine("\nStack:");
    Print(st);
```

Função Print

```
  private static void Print(IEnumerable tad) {
    Console.Write("[");

    foreach(var item in tad) {
      Console.Write(" {0}", item);
    }

    Console.WriteLine(" ]\n");
  }
```

14 – Crie uma função para inverter os dados da coleção recebida como parâmetro. Obs1: use qualquer outra estrutura que julgar necessária. **Obs2: não utilize o método reverse daclasse ArrayList.**

ArrayList

```
    ArrayList al = new ArrayList();

    for(int i=1; i<=10; i++) {
      al.Add(i);
    }

    Console.WriteLine("\nArrayList Normal:");
    Print(al);

    ReverseArray(al);

    Console.WriteLine("\nArrayList Reverse:");
    Print(al);
```

```
  private static void ReverseArray(ArrayList tad) {
    object?[] aux = tad.ToArray();
    tad.Clear();
    for(int i=aux.Length-1; i>=0; i--) {
      tad.Add(aux[i]);
    }
  }
```

Queue

```
    Queue qu = new Queue();

    for(int i=1; i<=10; i++) {
      qu.Enqueue(i);
    }

    Console.WriteLine("\nQueue Normal:");
    Print(qu);

    ReverseQueue(qu);

    Console.WriteLine("\nQueue Reverse:");
    Print(qu);
```

```
  private static void ReverseQueue(Queue tad) {
    object?[] aux = tad.ToArray();
    tad.Clear();
    for(int i=aux.Length-1; i>=0; i--) {
      tad.Enqueue(aux[i]);
    }
  }
```

Stack

```
    Stack st = new Stack();

    for(int i=1; i<=10; i++) {
      st.Push(i);
    }

    Console.WriteLine("\nStack Normal:");
    Print(st);

    ReverseStack(st);

    Console.WriteLine("\nStack Reverse:");
    Print(st);
```

```
  private static void ReverseStack(Stack tad) {
    object?[] aux = tad.ToArray();
    tad.Clear();
    for(int i=aux.Length-1; i>=0; i--) {
      tad.Push(aux[i]);
    }
```

Função Print

```
  private static void Print(IEnumerable tad) {
    Console.Write("[");

    foreach(var item in tad) {
      Console.Write(" {0}", item);
    }

    Console.WriteLine(" ]\n");
  }
```

15 – Crie uma função que receba a coleção como parâmetro e retorne a soma de seus elementos. Obs: considere que todos seus dados são do tipo int.

ArrayList

```
    ArrayList al = new ArrayList();

    for(int i=1; i<=10; i++) {
      al.Add(i);
    }

    Console.WriteLine("\nSoma Array: {0}\n", Soma(al));
```

Queue

```
    Queue qu = new Queue();

    for(int i=1; i<=10; i++) {
      qu.Enqueue(i);
    }

    Console.WriteLine("\nSoma Queue: {0}\n", Soma(qu));
```

Stack

```
      Stack st = new Stack();

      for(int i=1; i<=10; i++) {
        st.Push(i);
      }

      Console.WriteLine("\nSoma Stack: {0}\n", Soma(st));
```

Função Soma

```
    private static int Soma(IEnumerable tad) {
      int soma = 0;
      foreach(var item in tad) {
        soma += Convert.ToInt32(item);
      }
      return soma;
    }
```

16 – Crie uma função que calcule o número de elementos positivos de uma coleção passada como parâmetro.

ArrayList

```
      ArrayList al = new ArrayList();

      for(int i=-10; i<=10; i++) {
        al.Add(i);
      }

      Console.WriteLine("\nArray tem {0} números positivos\n",
ContaPositivos(al));
```

Queue

```
      Queue qu = new Queue();

      for(int i=-10; i<=10; i++) {
        qu.Enqueue(i);
      }

      Console.WriteLine("\nQueue tem {0} números positivos\n",
ContaPositivos(qu));
```

Stack

```
      Stack st = new Stack();

      for(int i=-10; i<=10; i++) {
        st.Push(i);
      }

      Console.WriteLine("\nStack tem {0} números positivos\n",
ContaPositivos(st));
```

Função ContaPositivos

```
    private static int ContaPositivos(IEnumerable tad) {
      int cont = 0;
      foreach(var item in tad) {
        int value = Convert.ToInt32(item);
        if (value > 0)
          cont++;
      }
      return cont;
    }
```

17 – Crie uma função que calcule o número de ocorrências em uma coleção de um elemento passado como parâmetro.

ArrayList

```
      ArrayList al = new ArrayList() {5, 5, 10, 15, 20, 25, 5, 20};

      Console.WriteLine("\nArray tem {0} números equivalentes a 5\n",
numOcorrencias(al, 5));
```

Queue

```
      Queue qu = new Queue();
      qu.Enqueue(5);
      qu.Enqueue(5);
      qu.Enqueue(10);
      qu.Enqueue(15);
      qu.Enqueue(20);
      qu.Enqueue(25);
      qu.Enqueue(5);

      Console.WriteLine("\nQueue tem {0} números equivalentes a 20\n",
numOcorrencias(qu, 20));
```

Stack

```
      Stack st = new Stack();
      st.Push(5);
      st.Push(5);
      st.Push(10);
      st.Push(15);
      st.Push(20);
      st.Push(25);
      st.Push(5);

      Console.WriteLine("\nStack tem {0} números equivalentes a 30\n",
numOcorrencias(st, 30));
```

Função numOcorrencias

```
    private static int numOcorrencias(IEnumerable tad, int value) {
      int cont = 0;
      foreach(var item in tad) {
        if (Convert.ToInt32(item) == value)
          cont++;
      }
      return cont;
    }
```