# Prática de Programação em Linguagem de Montagem-TP1

## 1 TRABALHO A SER DESENVOLVIDO E REGRAS DO JOGO

1. Implementar uma **função recursiva em linguagem de montagem do MIPS** para realizar pesquisa binária em um vetor sequencial pré-ordenado. O algoritmo básico de pesquisa binária a usar é dado abaixo, em C:

```
int BinSearch(const int A[], int Prim, int Ult, int Valor)
// -------------------------------------------------------
// Dado um vetor A, pesquisa entre elementos A[Prim]
//       até A[Ult] por Valor, usando pesquisa binária.
//
// Pré-condições: 0<=Prim, Ult<=Tam-1, onde Tam é o tamanho
// máximo do Vetor A e A[Prim]<=A[Prim+1]<=...<=A[Ult],
// ou seja, o vetor está ordenado em ordem crescente.
//
// Pós-condição: Se Valor existe no vetor A, retorna
// o índice do vetor igual a Valor, senão retorna -1.
// -------------------------------------------------------
{
   if (Prim > Ult)
      return -1;      // Valor não existe
   else
   {  // Invariante: Se Valor existe em A,
      //            A[Prim]<=Valor<=A[Ult]
      int Meio = (Prim + Ult)/2;
      if (Valor == A[Meio])
         return Meio;  // Encontrou Valor em A[Meio]
      else if (Valor<A[Meio])
         return BinSearch(A, Prim, Meio-1, Valor);
         // Recursão na metade inferior do vetor
      else
         return BinSearch(A, Meio+1, Ult, Valor);
         // Recursão na metade superior do vetor
   }  // end else
}  // end BinSearch
```

**Assuma que o conteúdo de A é: -5, -1, 5, 9, 12, 15, 21, 29, 31, 58, 250, 325 e teste pelo menos as seguintes chamadas:**

```
1. BinSearch(A, 0, 11, 325); // Deve retornar 11
2. BinSearch(A, 11, 11, -1); // Deve retornar -1
3. BinSearch(A, 0, 5, -1);   // Deve retornar 1
4. BinSearch(A, 4, 11, 31);  // Deve retornar 8
5. BinSearch(A, 0, 11, 17);  // Deve retornar -1
```

Defina uma área de dados adequada para o programa. Acrescente variáveis, se considerar necessário.

Respeite as seguintes convenções:

- Passagem de argumentos – todos os quatro (4) argumentos da função devem ser passados através da pilha, apontada pelo registrador $sp;
- O retorno do valor resultante da execução da função deve ser através do registrador $v0.

2. Elabore um programa principal para testar a rotina implementada. Lembre-se que este programa principal deve não apenas (1) chamar a rotina, mas também (2) passar argumentos para esta (como exigido acima, todos via pilha), (3) receber o valor de retorno, (4) imprimir este valor na console do simulador (utilizar a primitiva SYSCALL) e (5) esvaziar a pilha dos dados lá colocados. Note que os números são inteiros e não somente naturais. Assim, use as instruções de comparação do MIPS adequadas para lidar com números com sinal.
3. O trabalho deverá ser realizado em duplas e deve ser entregue até as 23hs e 59min do dia 23/abril/2019 via Moodle, na sala de entregas específica para este fim.
4. A dupla deve colocar os seus nomes como comentário no código fonte e renomear o arquivo conforme segue:

   NomeSobrenome1_NomeSobrenome2.asm.

5. A dupla deve fazer o upload do arquivo na sala de entregas do moodle até a hora estipulada. Apenas um aluno do grupo deve fazer o upload. Certifique-se que o seu trabalho está disponível na sala de entregas do moodle. Não serão aceitos trabalhos fora do prazo sob nenhuma hipótese.
6. Trabalhos copiados receberão nota ZERO.