Hérick Vitor Vieira Bittencourt, Lorina Zondervan e Luiz Augusto Inthurn

UNIVALI - Universidade do Vale de Itajaí

Ciências da Computação - Arquitetura e Organização de Processadores

Avaliação 1 - Programação em Linguagem de Montagem

Professor - Thiago Felski Pereira

Data *-* 10/04/2023

# 

[herick@edu.univali.br](mailto:herick@edu.univali.br), [zondervan@edu.univali.br](mailto:zondervan@edu.univali.br) e [luiz\_inthurn@edu.univali.br](mailto:luiz_inthurn@edu.univali.br)

# 

# 

# 

# 

# 

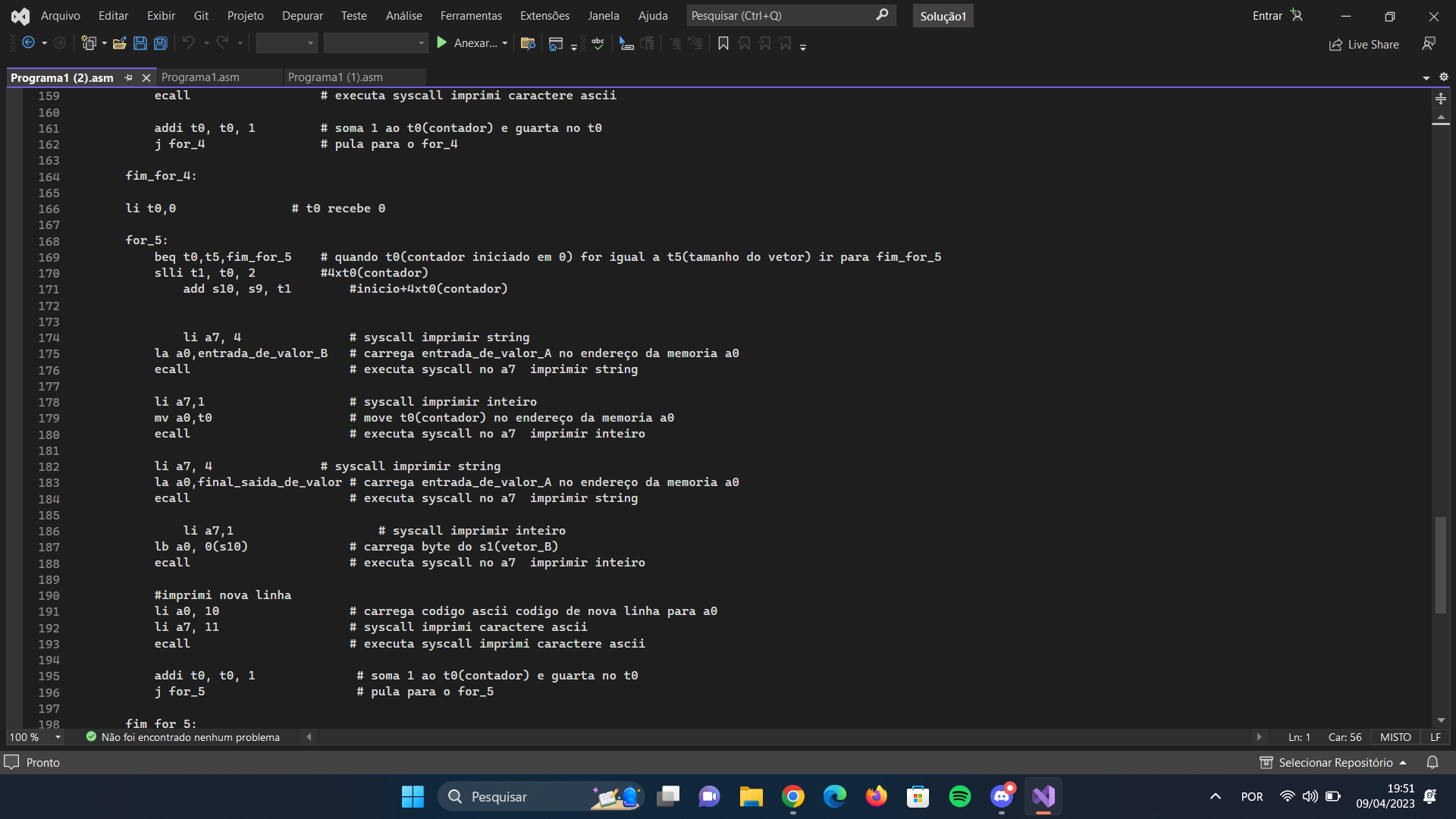
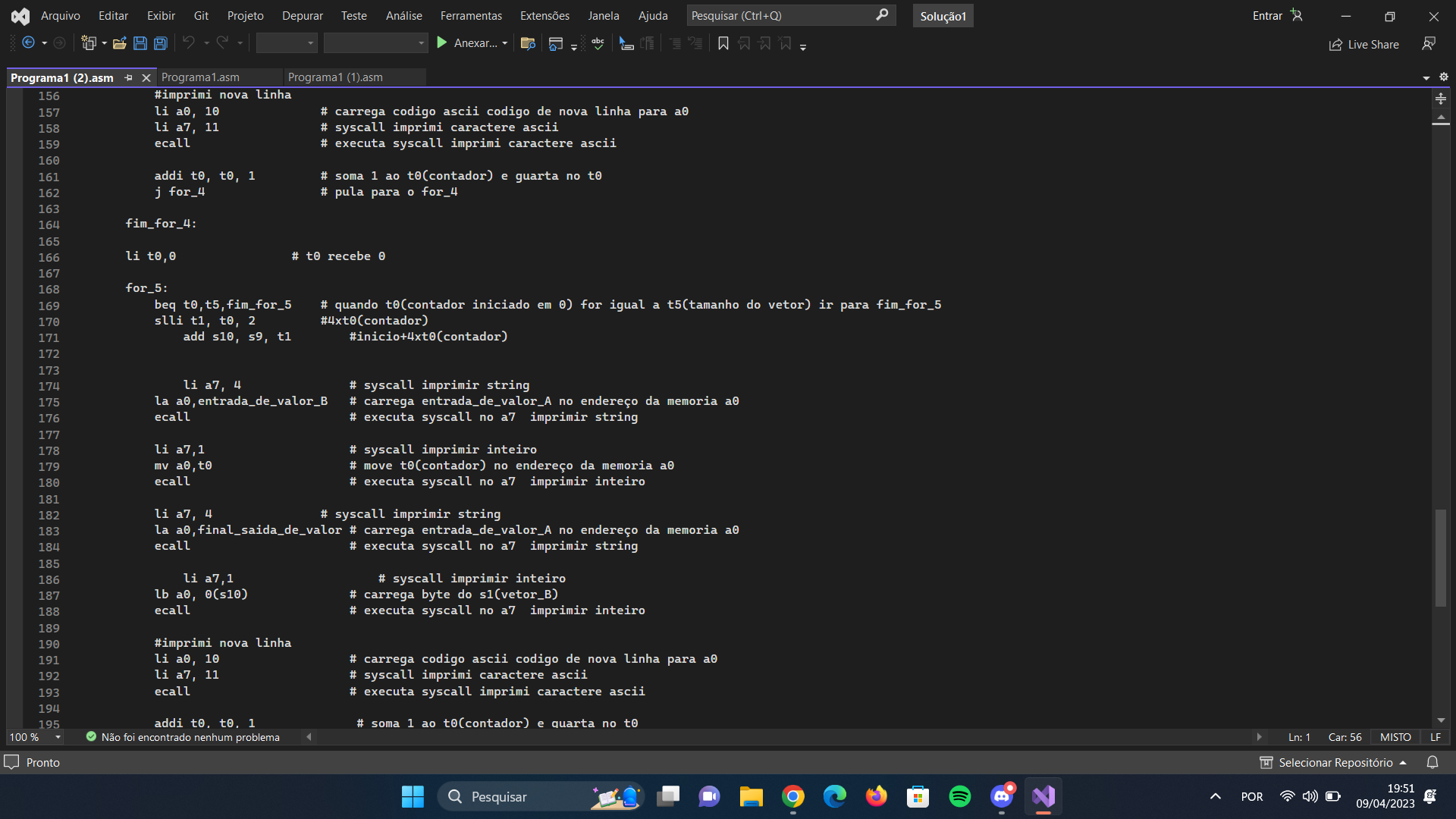
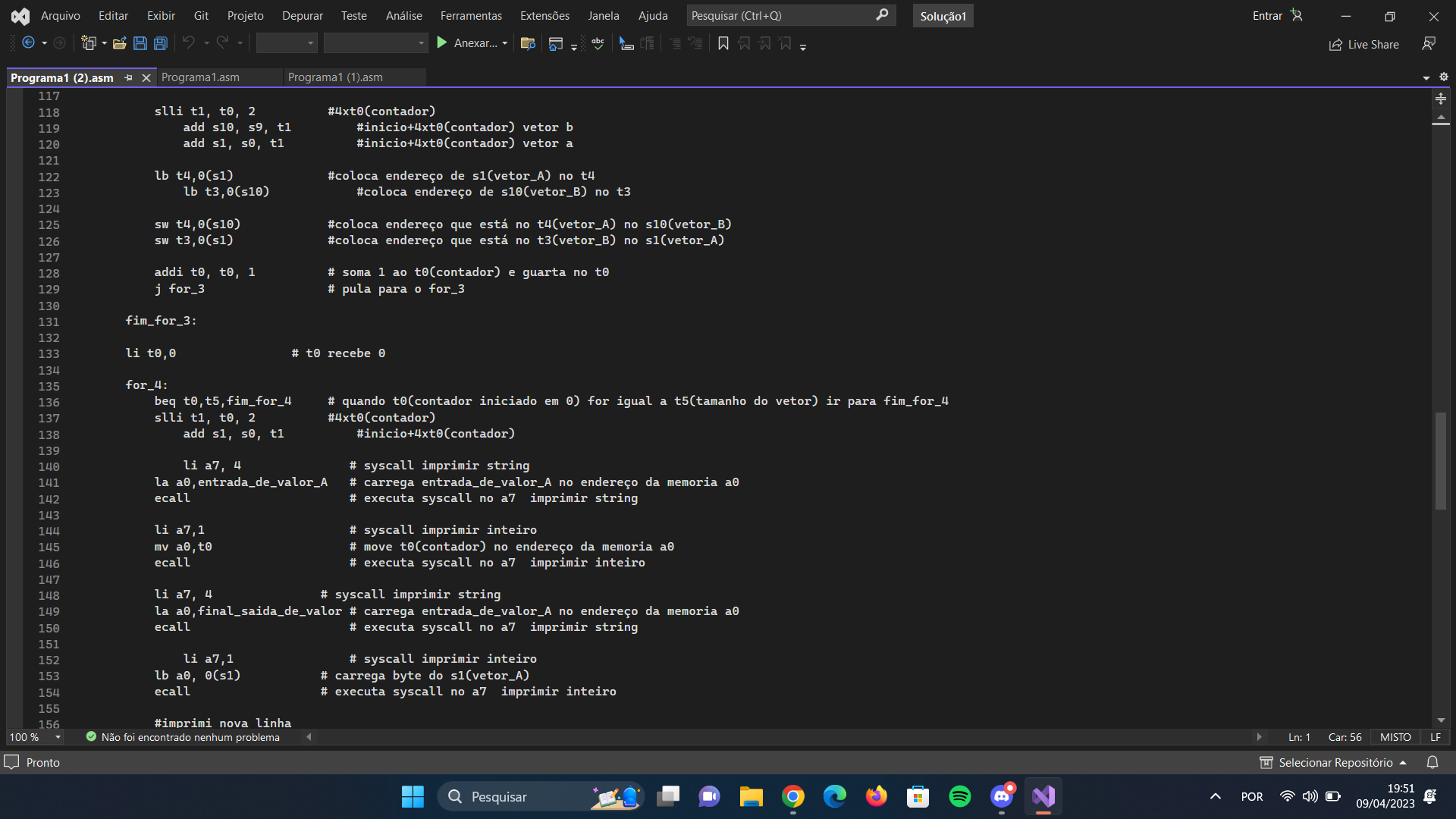
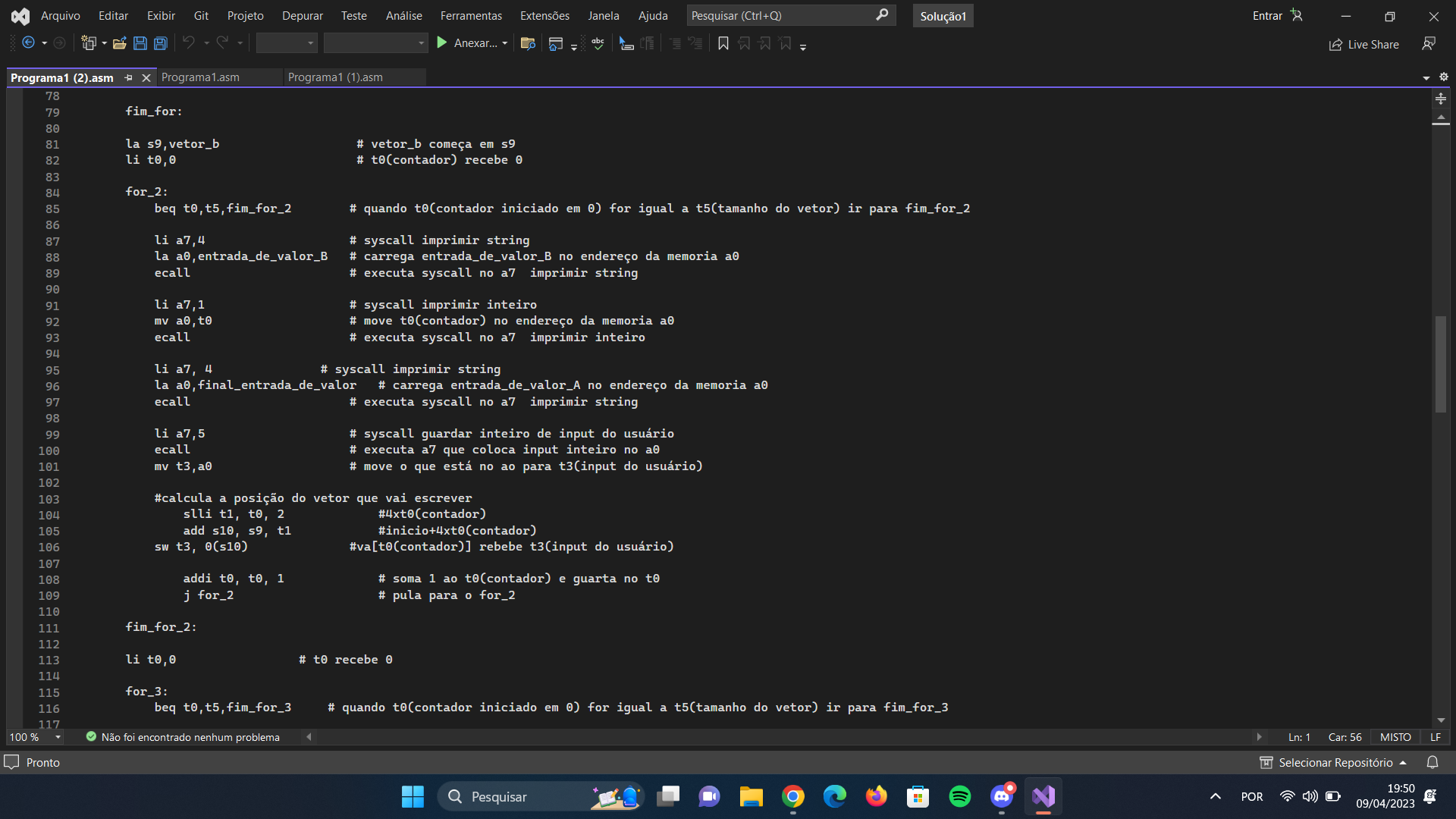
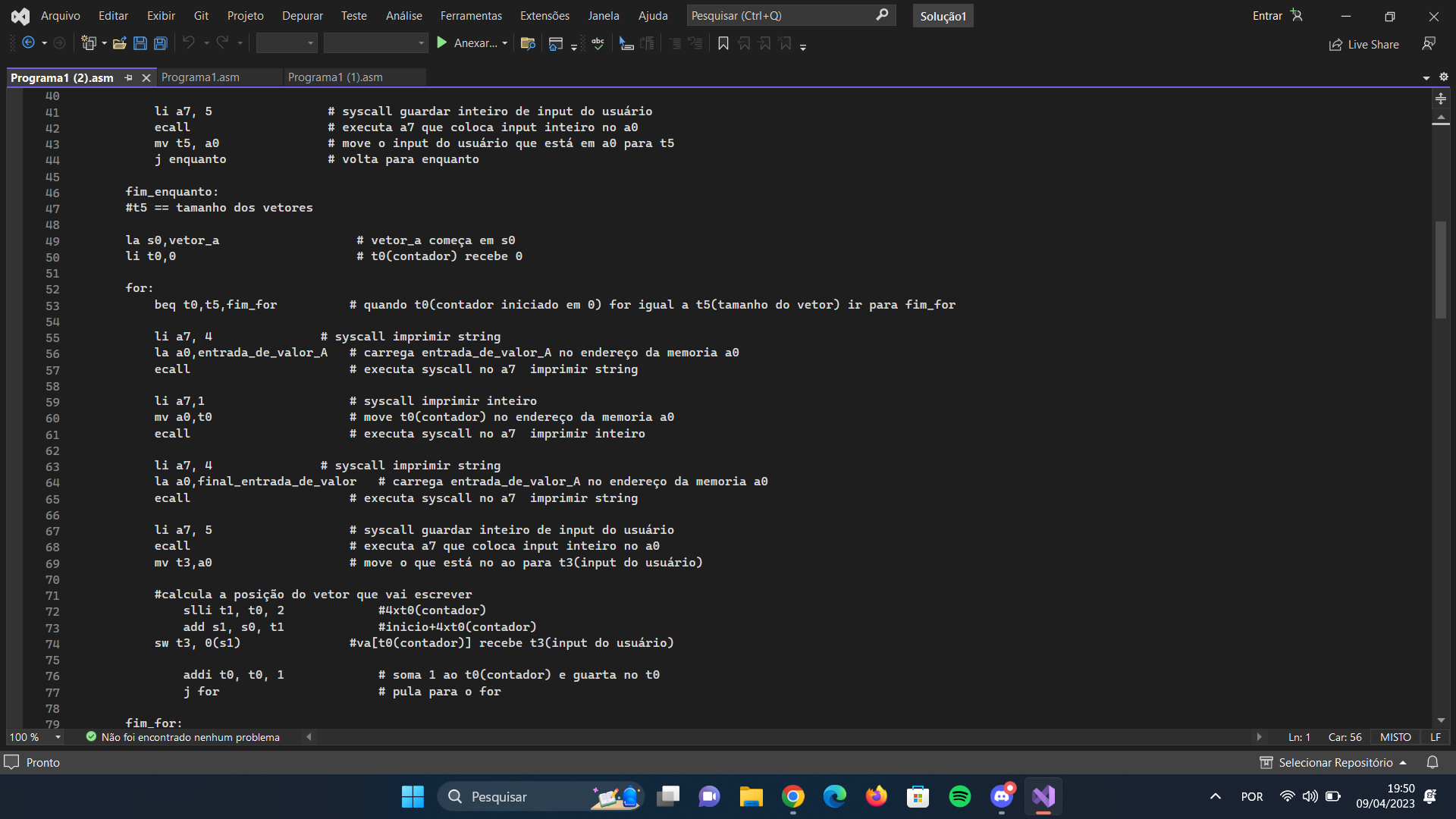
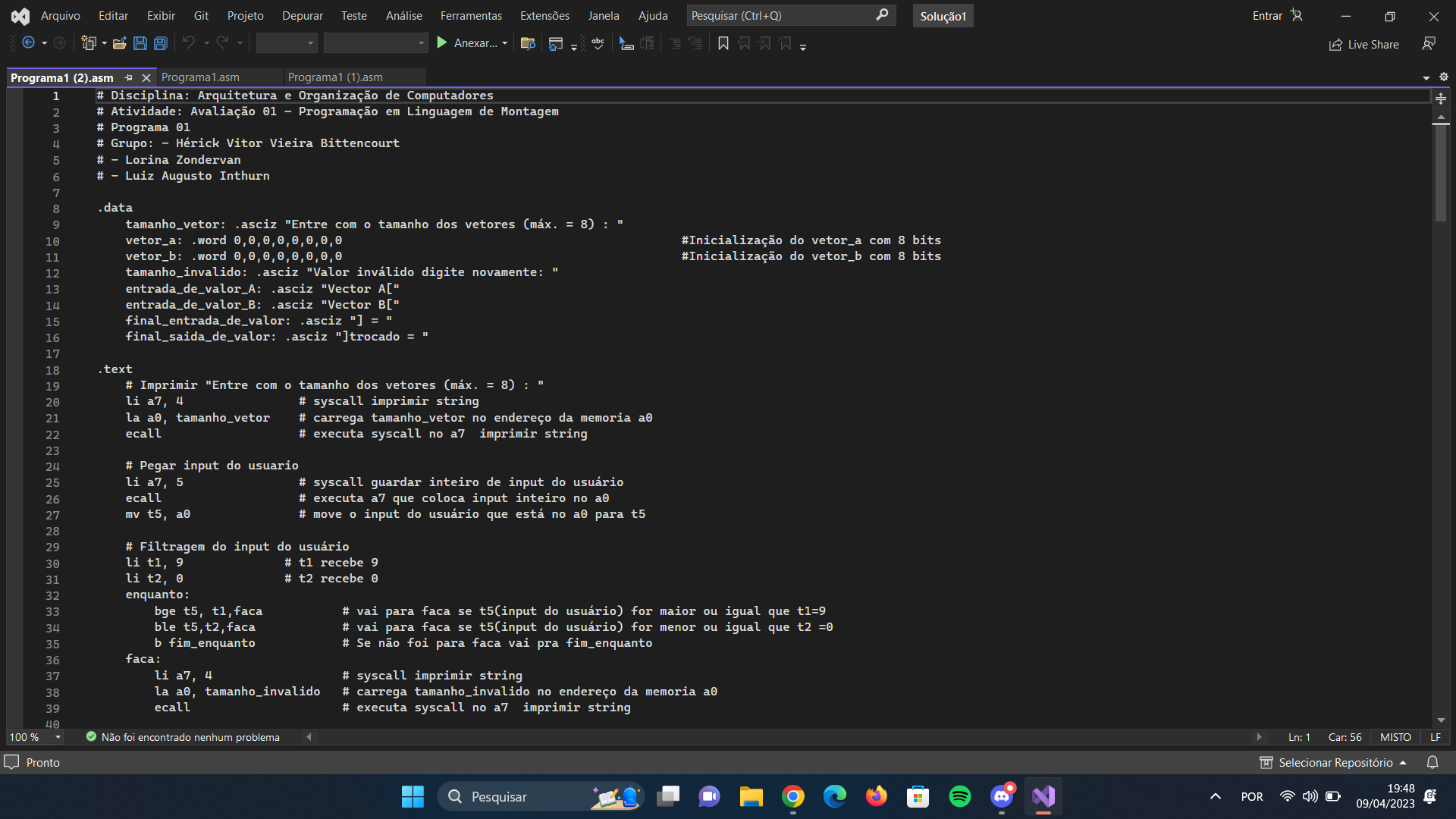
# 

# 

# Programa 1

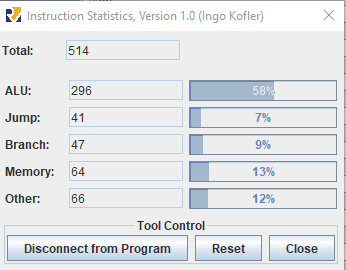
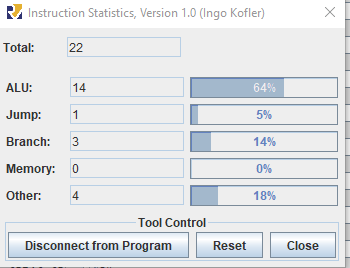
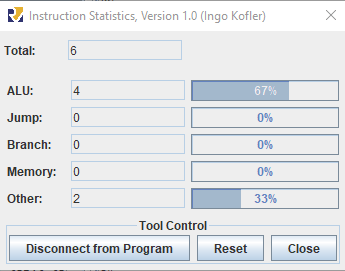
**Enunciado:** Implemente um programa que leia dois vetores via console e, após a leitura dos vetores, troque os conteúdos dos dois vetores. Por fim, o programa deve imprimir esse novo vetor na tela.

**Resolução:** Segue abaixo as capturas da resolução, do programa em funcionamento e instruction statistics e comentário do relatório.

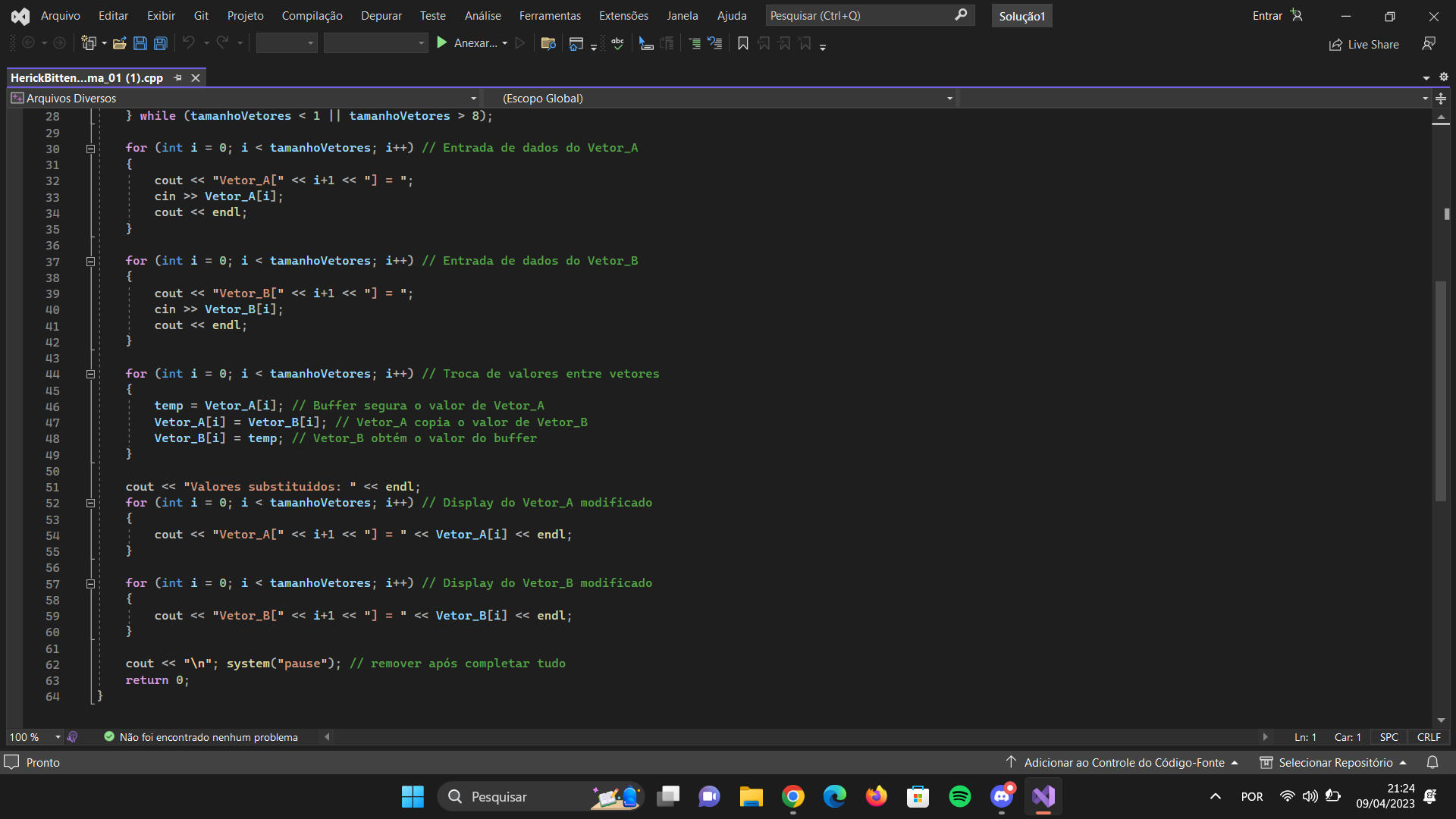
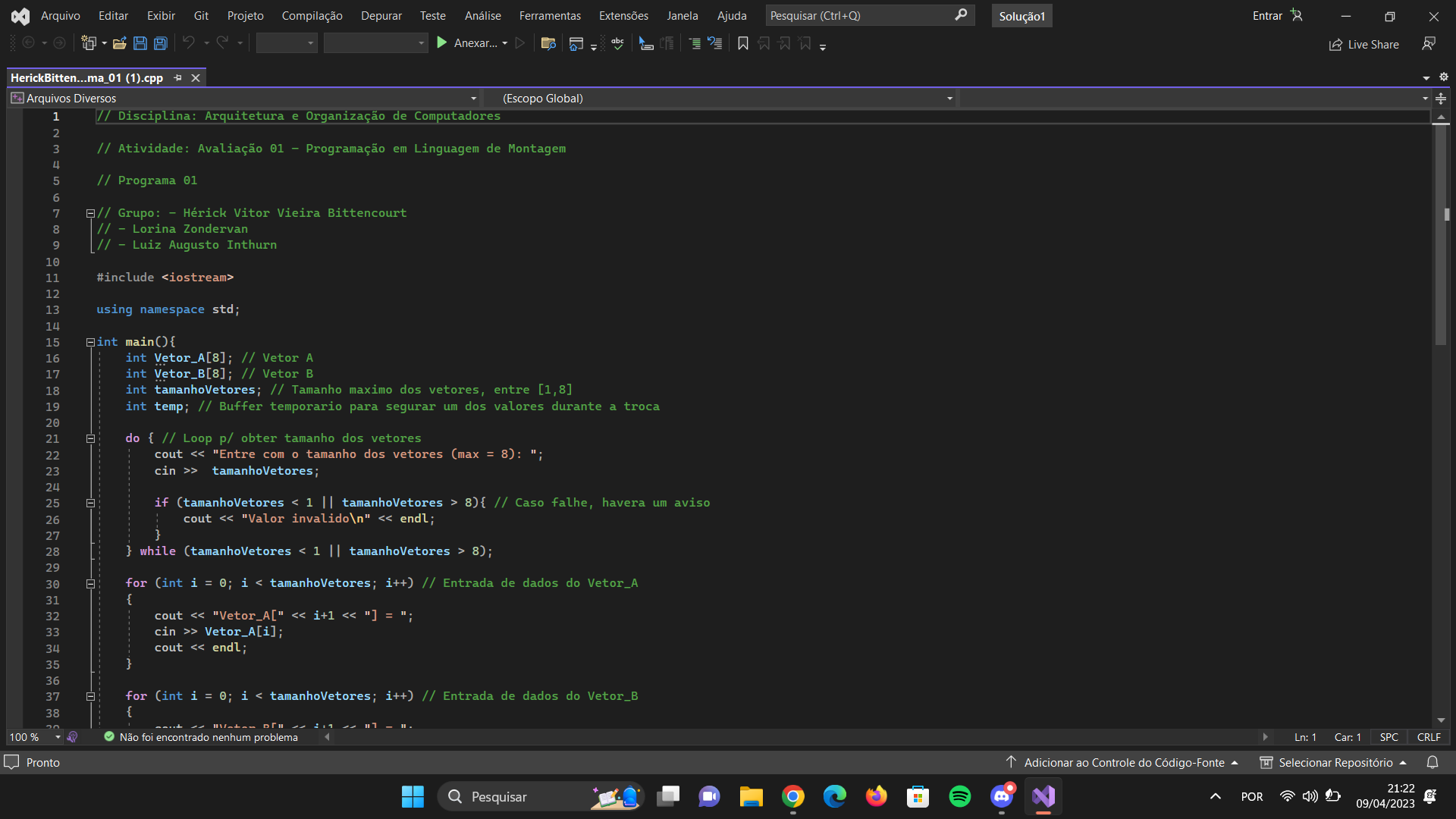


A seguir capturas do funcionamento do código: 

A seguir capturas do instruction statistics:



Código em C++:

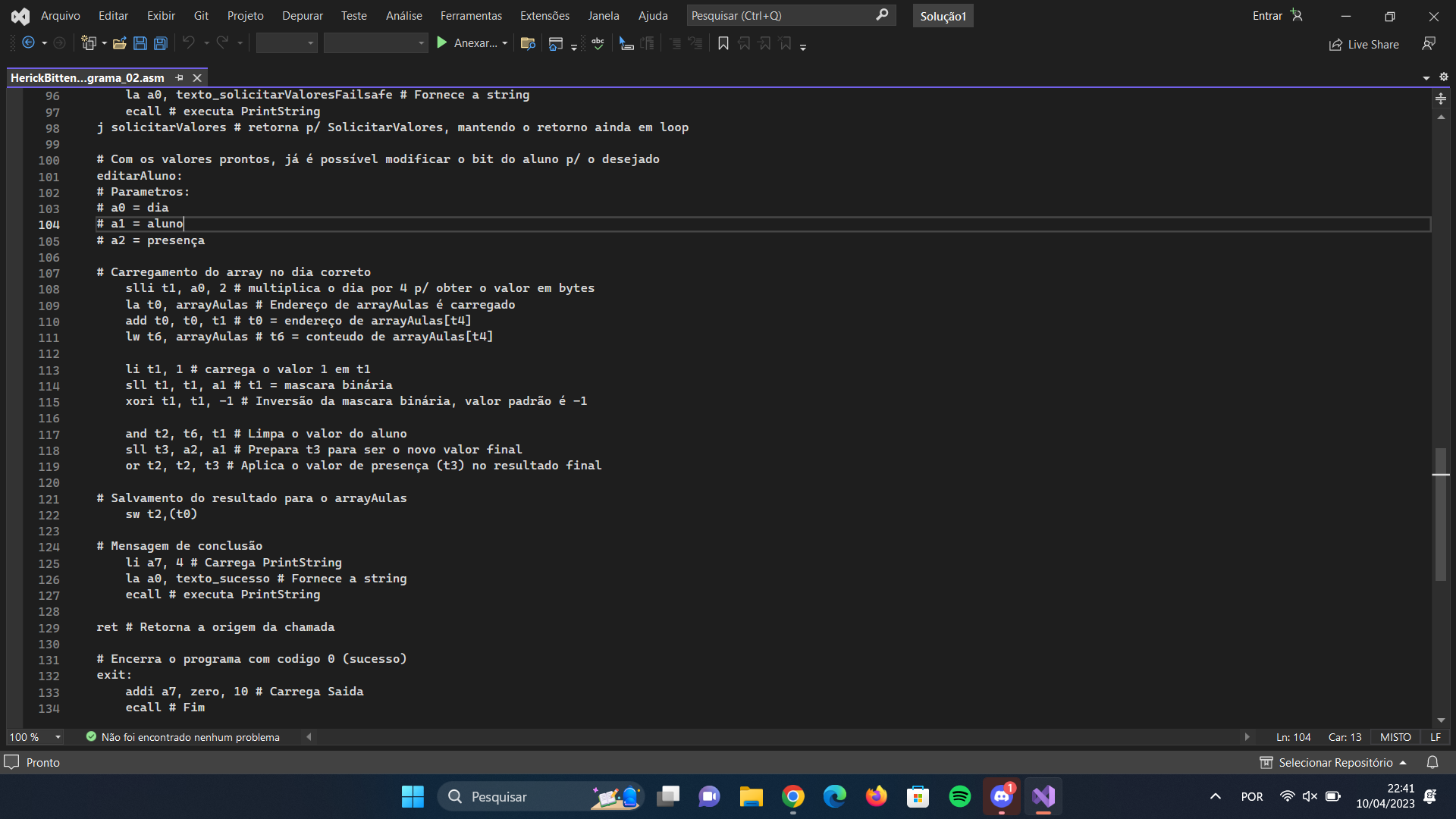
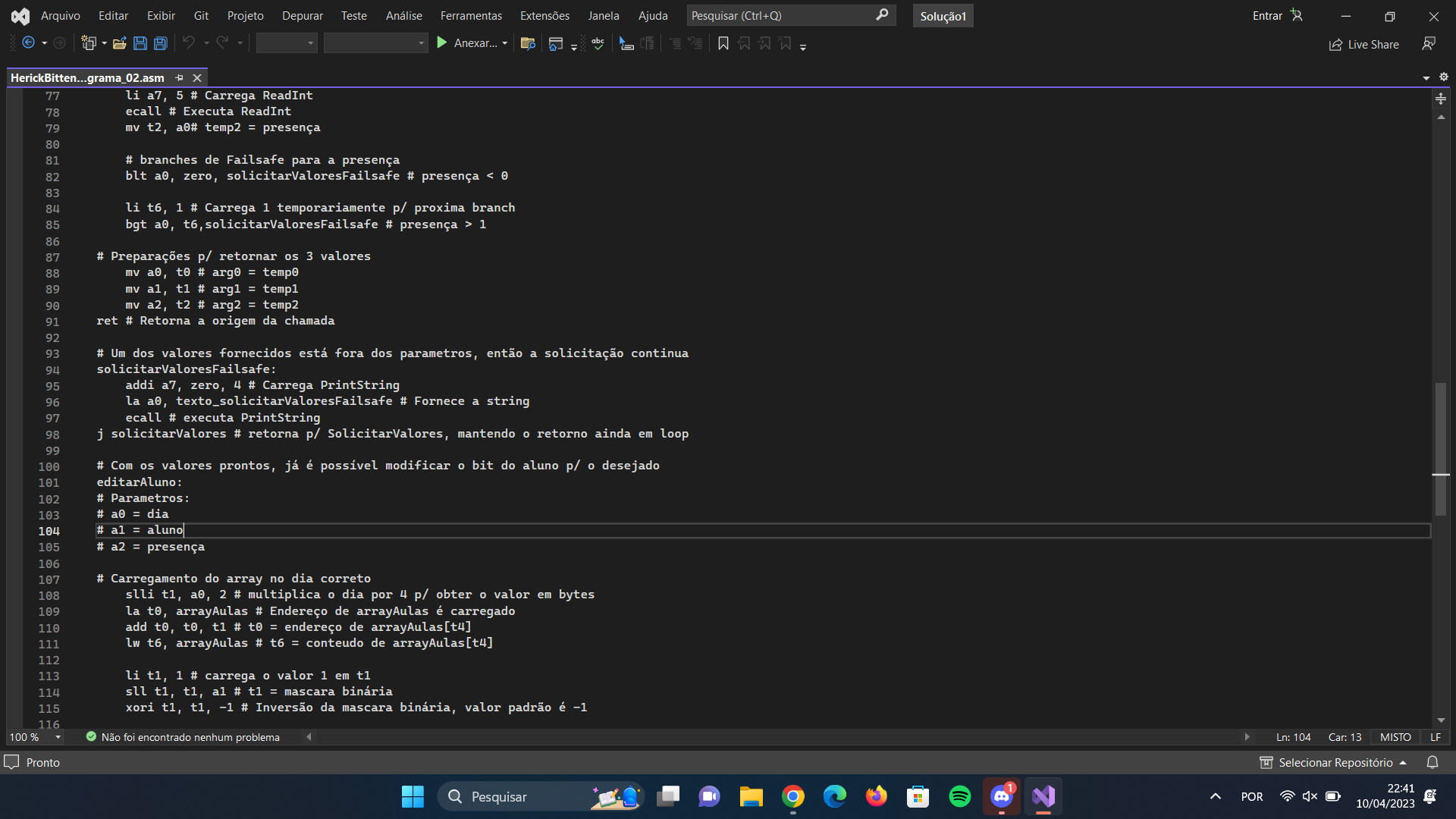
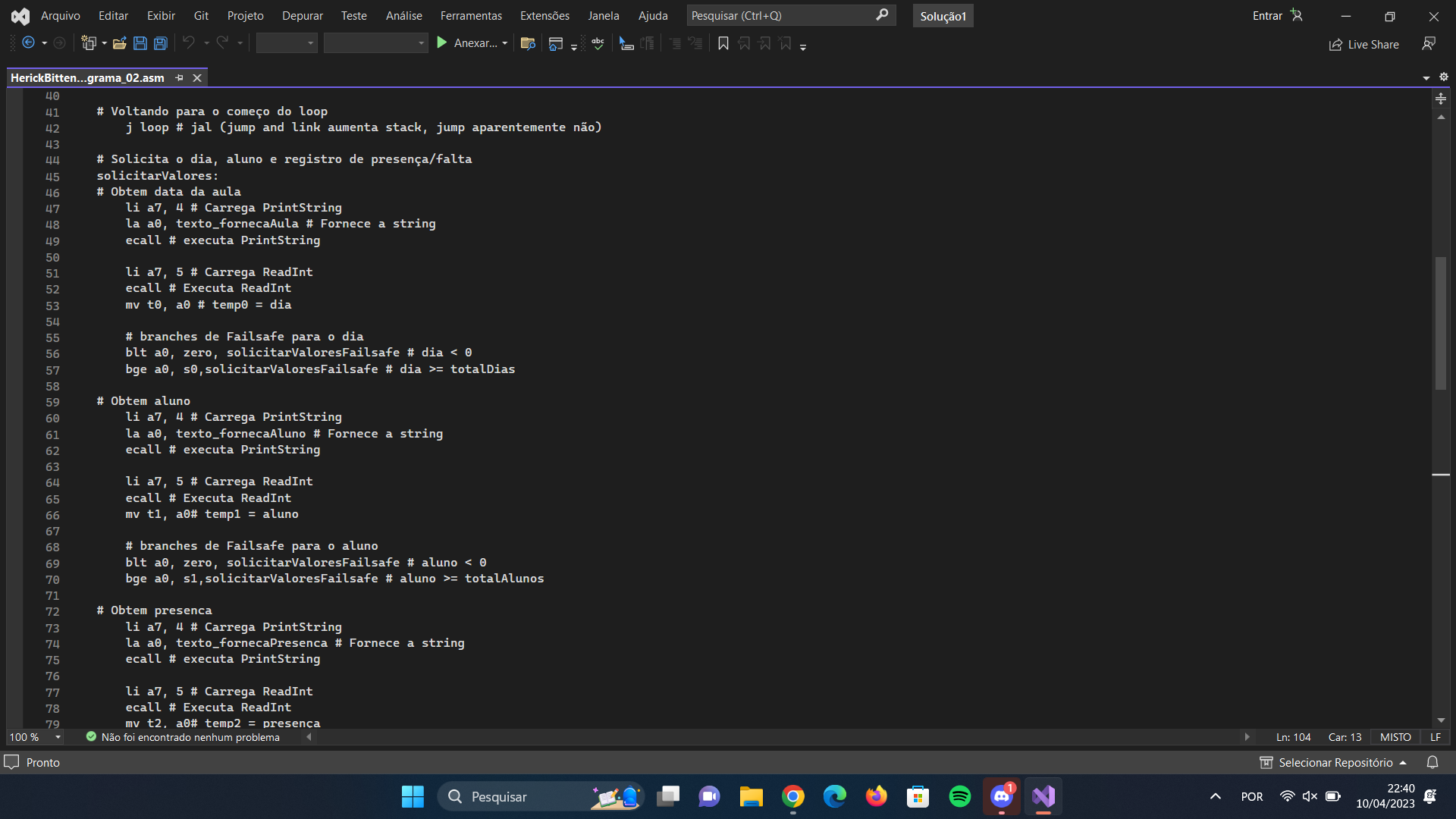
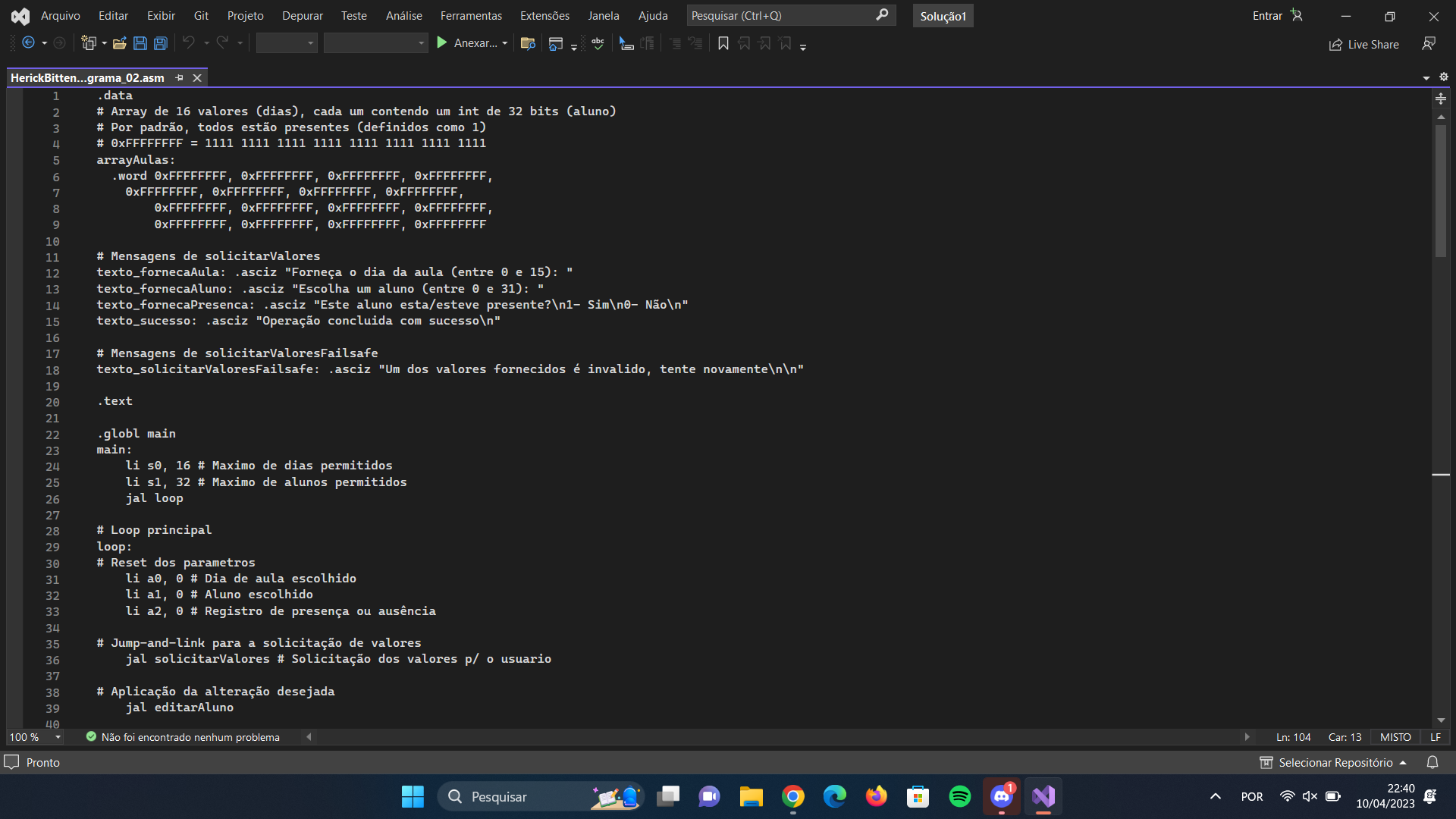


Podemos notar que o programa recebe dois vetores de tamanho que é definido pelo usuário e imprime os mesmos, trocando os valores entres eles. Sendo assim, vamos partir para o começo do código, O programa começa declarando algumas variáveis na seção “.data”, como a string “Entre com o tamanho dos vetores (max. = 8):” e os dois vetores “vetor\_a” e o “vetor\_b”. Logo em seguida, na seção “.text”, começa a execução do programa. Com isso esclarecido podemos chegar na primeira parte do programa onde o usuário é solicitado a inserir o tamanho dos vetores, que por sua vez é armazenado na variável “t5”. Logo em seguida, chegamos em um loop que vai verificar se o tamanho inserido é válido, ou seja, se está de acordo entre 1 e 8. Caso contrário, o programa imprime a mensagem “Valor inválido, digite novamente:” e pede ao usuário que insira um novo valor. Esse processo é repetido até que um dos valores que for inserido seja válido. Agora chegamos na segunda parte do programa onde o usuário vai ser solicitado para inserir os valores de cada elemento do vetor “vetor\_a”, um de cada vez. Para cada elemento, o programa imprime a mensagem “Vector A[i] = “ (onde o “i” é o elemento de índice do elemento), solicita que o usuário insira o valor do elemento e armazena esse valor no vetor “vetor\_a”. Sendo assim após a entrada dos valores do “vetor\_a”, o programa segue um processo semelhante para preencher o “vetor\_b” com os valores inseridos pelo usuário. Por fim chegamos na última parte do programa, onde o conteúdo dos vetores “vetor\_a” e “vetor\_b” é trocado, elemento por elemento, Para cada elemento “i” dos vetores, o programa imprime as mensagens “Vector A[i] trocado = “ e “Vector B[i] = “ respectivamente, e então troca o valor dos elementos nos dois vetores. Resumindo o programa que foi feito para responder o enunciado 1(um) é um programa que foi feito em linguagem de montagem que solicita ao usuário dois vetores de tamanho definido e imprime os vetores trocando os valores de seus elementos. O programa inclui verificações para garantir que o tamanho do vetor inserido pelo usuário seja válido.

**Programa 2**

**Enunciado:** Considere uma disciplina com 16 dias de aulas e que é necessário registrar a presença de cada aluno em cada dia, consumindo o mínimo de variáveis em memória. Assumindo que a turma tenha no máximo 32 alunos, implemente um programa que utilize um vetor de 16 elementos inteiros (um elemento para cada dia) para registrar a frequência dos alunos de cada disciplina e que ofereça ao professor funções para registrar a presença e ausência de qualquer aluno em qualquer dia.

**Resolução:** Segue abaixo as capturas da resolução, do programada em funcionamento e instruction statistics e o comentário do relatório.



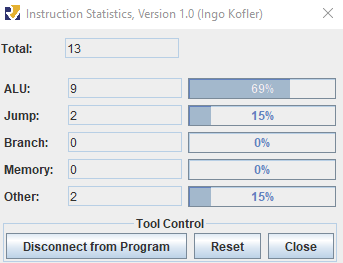
Podemos notar que a resolução do segundo problema dado foi criar um código onde é simulado a presença de alunos em uma escola ou universidade, por meio de um array de 16(dezesseis) valores (dias), cada um contendo um valor de 32(trinta e dois) bits para cada aluno totalizando 32(trinta e dois) alunos.

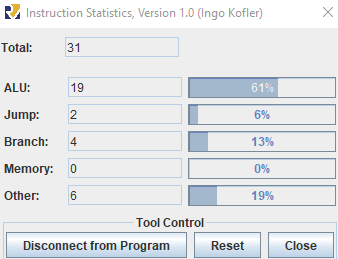
O programa inicia definindo o array de aulas, onde vemos que todos os alunos já começam com presenças e definidos como 1(um). Em seguida, vemos que são definidas as mensagens de solicitações de valores que o programa solicita ao usuário, caso ele queira alterar a presença de algum aluno. O programa define a função principal (main) e inicia um loop, que será o responsável por solicitar ao usuário os valores necessários para a alteração na presença dos alunos, aplicar a alteração e, em seguida, reiniciar o loop.

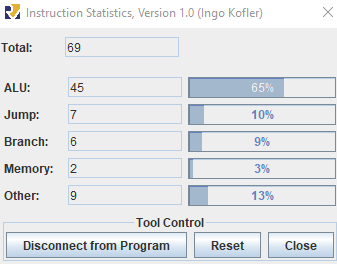
Chegando dentro do loop principal, a função solicitarValores é chamada, e a partir dela o programa solicita ao usuário o dia, o aluno e a presença desejada. Após a validação dos valores fornecidos pelo usuário se estão dentro dos limites permitidos, a função editarAluno é chamada. Essa função é responsável por alterar o bit correspondente ao aluno no array de aulas, indicando sua presença ou porventura a sua ausência. Podemos ver que caso o usuário informe algum valor inválido, o programa executa a função solicitarValoresFailsafe, que informa que um dos valores fornecidos é inválido e retorna para a função solicitarValores. Chegando no final do código podemos notar que é implementado duas funções principais que são elas solicitarValores e editarAluno com isso vemos que a função solicitarValores fica responsável por solicitar ao usuário três valores sendo eles dia da aula, aluno escolhido e o registro de presença ou ausência. Essa função utiliza a syscall Readint para obter valores do usuário e, em seguida, faz algumas verificações para garantir que os valores informados estejam dentro dos limites permitidos. Caso algum dos valores esteja fora do limite, a função solicitarValoresFailsafe é chamada para informar ao usuário que algum valor inválido foi informado e a solicitação contínua. Caso contrário, a função retorna os três valores obtidos para a função chamadora. Seguindo adiante vemos que a função editarAlunos é responsável por modificar o bit de presença e a ausência do aluno selecionado no dia de aula correspondente. Para isso, ela usa o valor do dia de aula informado para conseguir calcular o índice do arrayAulas correspondente ao dia escolhido e logo em seguida, ela utiliza uma máscara de bits para resetar o bit correspondente ao aluno, seguido por uma operação lógica OR, cujo define o bit para o valor desejado de presença ou ausência. Sendo assim vemos que o código termina com um jump incondicional para o início do loop principal, garantido que o programa consiga continuar solicitando os valores e modificando os bits de presença e ausência do arrayAulas enquanto o usuário desejar continuar.

A seguir capturas do instruction statistics:

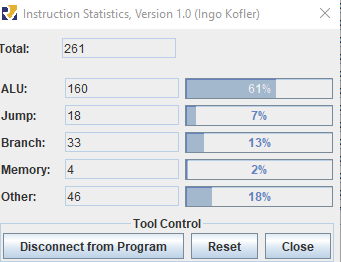
Loop concluído sem erros:







Uma operação sucedida e 9 failsafes (todas as branches acionadas):



Código em C++: