Trabalho Prático 2
Software Básico
Prof. Bruno Macchiavello

1 Introdução
O trabalho consiste em implementar em C/C++ um método de tradução de uma linguagem de máquina simples para linguagem IA-32.

O trabalho pode (e recomenda-se) ser feito em grupo de 2 alunos. Deve ser entregue somente o código e um arquivo README indicando o SO utilizado e como compilar o programa. Sendo que se for LINUX deve-se utilizar o GCC, se for Windows deve ser o CODEBLOCKS. Apple OS será tratado como programa em LINUX.

2 Objetivo
Fixar o funcionamento de um processo de tradução. Especificamente montagem e ligação.

3 Especificação

3.1 tradutor

O trabalho deve traduzir um código objeto em linguagem hipotética vista em aula para assembly IA-32. O arquivo de entrada vai ser idêntico ao arquivo de saída do Trabalho 1. Sendo que estamos falando do arquivo que já está ligado, ou de um código que não precisa ser ligado. Ou seja de uma sequência de números numa mesma linha, exemplo:

12 29 10 29 4 28 11 30 3 28 11 31 10 29 2 31 11 31 13 31 9 30 29 10 29 7 4 14 2 0 0 0

Esse código deve ser traduzido para um código de ASSEMBLY IA-32 (não máquina). Vamos assumir que:

- Estamos trabalhando TUDO em 32 bits. Ou seja, os espaços de memória, o acumulador, o contador de programa do assembly inventado é tudo 32 bits.
- Os números TEM sinal
- A multiplicação e divisão devem ser traduzidas utilizando MULTIPLICAÇÃO e DIVISÃO em IA-32 (não shifts, somas e subtrações).
- Na multiplicação, o código deve identificar se deu OVERFLOW. Para a execução do programa e indicar que aconteceu overflow.
- Assumir que nunca tem erro dos arquivos de entrada.
- Não é necessário que tenha UM registrador específico para ser o acumulador do assembly inventado.

Serão criada DOIS opcodes a mais do assembly inventado: S_INPUT e S_OUTPUT. Ambos recebem 2 argumentos. O primeiro é um LABEL de um endereço de memória que tenha sido reservado utilizando SPACE com a quantidade de espaços de memória necessários. O segundo argumento deve ser a quantidade de caracteres a serem lidos ou escritos. Ou seja no assembly inventado seria algo como:

```
S_INPUT LABEL1, 5
.
.
.
LABEL1: SPACE 5
```

O OPCODE das instruções serão 15 para S_INPUT e 16 para S_OUTPUT. Ambos utilizando dos argumentos no código objeto também o primeiro representando a contagem do LABEL e o segundo o número indicado, no exemplo seria a contatem da LABEL1 e o número 5.

O tradutor deve ser feito em linguagem C ou C++. Deve receber o arquivo de entrado por argumento, ex: ./tradutor meuarquivo.obj. A saída deve ser o mesmo nome do arquivo com extensão ".s".

Porém, parte do trabalho deve ser diretamente implementado em IA-32. Quando tiver código de INPUT, OUTPUT, S_INPUT ou S_OUTPUT. O código em C deve substituir por uma chamada a função em IA-32. As funções devem ser copiadas no início do código em IA-32 (as funções podem ser copiadas mesmo sem ser utilizadas). Estas funções devem receber argumentos pela PILHA, se usarem variáveis locais devem ser criadas na pilha também. As funções devem implementar chamadas ao sistema de forma de ler/escrever números e strings. Assumir que os números têm a quantidade máxima de dígitos de um número de 32 bits com sinal (lembrar que o sinal conta como um caractere digitado). Neste caso devem ser traduzido os labels utilizados no assembly inventado para os opcodes das instruções de I/O como arrays de bytes e não arrays de 32 bits.

As funções devem devolver a quantidade de bytes lidos/escritos no EAX. E imprimir a mensagem de quantidade de bytes lidos/escritos sempre que utilizadas. Ex: deve aparecer na tela, "Foram lidos/escritos XXX bytes".