Trabalho Prático 1
Software Básico
Prof. Bruno Macchiavello

1 Introdução

O trabalho consiste em implementar em C/C++ um montador e ligador do assembly inventado visto em sala de auala.

O trabalho pode (e recomenda-se) ser feito em grupo de 2 alunos. Deve ser entregue somente o código e um arquivo README. O arquivo README deve indicar os nomes dos alunos, SO utilizado, como compilar o programa e como rodar (a princípio seguir a indicação da especificação). 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. Recomenda-se rodar em 2 computadores diferentes antes de enviar o trabalho.

Não é permitido o uso de bibliotecas ADICIONAIS. Pode ser utilizado qualquer padrão de C ou C++. Somente UM dos alunos da dupla deve enviar o trabalho pelo APRENDER.

2 Especificação

2.1 Montador

O montador deve ser chamado de MONTADOR e ter 2 modos de execução. Ao rodar "./montador -p myfiile.asm" o montador deve gerar um arquivo pre-procrssado myfile.pre. Ao rodar "./montador -o myfile.pre" deve gerar um arquivo myfile.obj. o montador nao deve ser sensível ao caso podendo aceitar maiusculas e minúsculas em qualquer ligar do código. Deve poder aceitar espaços em branco e tabulações em qualquer lugar do código. Tambem deve aceitar quebra de linhas entre rótulo e operação. No comando copy os argumentos estarão separados por vírgula SEM espaço. O código tb deve aceitar comentário em qualquer parte do código, o comentário é indicado pelo caracter ";".

2.1.1 Pre processamento

O arquivo de entrada pode ter EQU e IF. As diretivas EQU sempre ficaram nas primeiras linhas do codigo. Ja as diretivas IF podem estar em qualquer lugar. O pre processador deve avaliar e retirar essas diretivas conforme visto em sala de aula. Reforçando novamente o mantador deve:

- Aceitar Maiúsculas e Minúsculas (não ser sensível ao CASO)
- A diretiva CONST deve aceitar positivos, negativos e hexa no formato 0x (ex: 0x12). .
- O comando COPY deve separar os argumentos por "," SEM espaço

- Desconsiderar todos os espaços, tabulações ou enter desnecessários.
- Pode dar rótulo seguido de dois pontos e ENTER. O rótulo é considerado como da linha seguinte
- Aceitar comentário em qualquer parte do código iniciado por ; (o comentário deve ser removido no pré-processamento de EQU e IF)
- Assumir que os SPACES E CONST sempre estao no final

A saida do pre processador deve ser um arquivo de texto SEM comentários, EQU e IF.

2.1.2 montagem

O montador deve operar de 2 formas diferentes caso o programa tiver BEGIN / END ou nao. Se nao tiver “begin e end” o programa deve entregar como saida um arquivo de TEXTO sendo uma única linha do programa montado. Na diretiva SPACE deve ser usado 00 e nao XX. Olhar o exemplo do arquivo OBJ no Moodle. Essr arquivo deve ser capaz de rodar no simulador tb disponível mo moodle. O montagem DEVE ser realizada usando o algoritmo de PASSAGEM UNICA.

Caso o arquivo de entrada tenha begin / end deve entregae um arquivo OBJ com tabela de uso, tabela de diretivas e informacao de realocacao. Confirme exemplo do Moodle.

O montador deve ser capaz de detectar as seguintes erros:

- Rotulo ausente
- Rotulo dobrado na mesma linha
- Rotudo redefinido
- Numero de operandos errada p uma instrucao
- Erros lexicos de rotulos (rotulos nao podem começar com número e unico caracter especial q podem ter eh “_”.)
- Instrução ou diretiva inválida

2.2 Ligador

O ligador deve chamar LIGADOR. Deve ser usado “./ligador prog1.obj prog2.obj”. Deve dar como saida um arquivo chamado “prog1.e”. A entrada do ligador deve ser os arquivos obj gerados pelo seu montador em caso de ter achado diretivas BEGIN e END. O arquivo de saida “.e” deve ser um arquivo de texto de uma unica linha igual ao OBJ q nao precisa der ligado. Deve ser capa de rodar no simulador. O ligador funciona no máximo com 2 arquivos unicamente.