Curso: Bacharel em Sistemas de Informação **Turma:** BSI2
Disciplina: Arq. e Organização de Computadores **CH:** 60 horas
Período: 2018/1
Professor: Flávio Giraldeli

Resolvendo problemas em C e Assembly

Objetivo

Implementar três pequenos programas usando a linguagem **Assembly x86** (16 bits) e **C**. Para cada caso, após a implementação em alto e baixo nível, uma análise comparativa deverá ser feita a fim de evidenciar os detalhes de Arquitetura de Computadores que são ocultados pela linguagem de alto nível, C.

Grupos

Deverão ser formados grupos com 5 alunos, a serem escolhidos livremente (contando que pelo menos um domine os aspectos básicos de programação em C). No caso de o número de alunos da turma não ser múltiplo de 5, a decisão ficará a critério do professor.

Pontuação

O referido trabalho será avaliado de 0 a 100 e corresponderá a 20% da nota semestral. A nota do trabalho será calculada da seguinte forma:

$$Nota\ do\ Trabalho = Implementações + 0{,}4 * Relatório$$

O relatório será pontuado de 0 a 100 e terá peso 40%. As implementações também serão avaliadas de 0 a 100 e possuem os seguintes pesos: Q1 (10%), Q2 (20%) e Q3 (30%). E, em cada questão, a nota é composta por 75% (Assembly) + 25% (C). Atenção para o peso de cada questão e o fato da implementação em Assembly contribuir muito mais para a nota que a correspondente em C.

Não serão admitidos, em hipótese alguma, trechos de código comuns ou semelhantes entre grupos. Isto será rigorosamente observado. Assim sendo, evitem a "cooperação" entre os grupos. A mesma observação vale "pesquisas na internet". Lembre-se, o professor também sabe fazer essas pesquisas.

Da Implementação

Os três problemas abaixo foram cuidadosamente propostos para explorarem aspectos básicos da programação em baixo nível. Cada um deles deverá ser resolvido separadamente, com arquivos com a seguinte nomenclatura:

- `Problema1.c` `Problema1.asm`
- `Problema2.c` `Problema2.asm`
- `Problema3.c` `Problema3.asm`

Em cada uma das implementações, o comportamento do programa executável deverá ser indistinto, ou seja, em execução, tanto o executável originado do código C, quanto o provindo do Assembly têm que se comportar **exatamente da mesma forma** do ponto de vista do usuário.

A implementação em C poderá ser feita no compilador que o grupo desejar (ou que já esteja acostumado a usar). Ou, se quiser uma dica, pode começar a usar uma IDE profissional como o ***Microsoft Visual Studio Community 2017*** (Gratuito). (Link: https://www.visualstudio.com/pt-br/products/visual-studio-community-vs)

Já a implementação em Assembly deverá fazer uso do **emu8086**, enviado por email. O formato do arquivo Assembly poderá ser um .com, cuja estrutura "mono-segmento" é bem mais simples que um .exe. Nenhum procedimento/macro pré-pronto poderá ser chamado diretamente. No entanto, você poderá abrir o arquivo inc\emu8086.inc e copiar uma ou outra rotina básica e inserir diretamente no seu código. As rotinas (macro/procedimento) deverão ser traduzidas e plenamente compreendidas pelo grupo. A qualidade da sua implementação, usando corretamente os recursos aprendidos (como chamadas a procedimento, macros, uso da pilha, saltos, variáveis, etc...) será avaliado.

Faça uso extensivo do emulador. Lembre-se que o objetivo primário é o aprendizado de AOC, e não a programação em Assembly. O emulador possui simulação passo a passo da execução, sendo possível visualizar todas as informações de estado (registradores, memória, pilha, variáveis, etc...) em tempo real.

Os Problemas

Requisitos: Comparações, Desvios condicionais, operações aritméticas, input/output de caracteres.

1) Volte a sua disciplina de Programação 1[1]. Olhe os primeiros problemas que você fez. Encontre um que use apenas comandos de decisão e operações aritméticas (ou seja, nada de loops e nem de funções). **Especifique o problema no relatório** e o resolva em C/Assembly conforme as especificações desse trabalho. Mas, atenção: A escolha de um exercício trivial fará com que sua pontuação seja significativamente mais baixa. #DeGraçaEssa

   Requisitos: Anterior + *Loops*.

2) O método de ordenação de vetores conhecido como *Insertion Sort* é um método simples e conhecido, apesar de algumas ineficiências. Podemos fazer uma comparação do *Insertion Sort* com o modo de como algumas pessoas organizam um baralho num jogo de cartas. Imagine que você está jogando cartas. Você está com as cartas na mão e elas estão ordenadas. Você recebe uma nova carta e deve colocá-la na posição correta da sua mão de cartas, de forma que as cartas obedeçam a ordenação. A cada nova carta adicionada à sua mão de cartas, a nova carta pode ser menor que algumas das cartas que você já tem na mão ou maior, e assim, você começa a comparar a nova carta com todas as cartas na sua mão até encontrar sua posição correta. Você insere a nova carta na posição correta, e, novamente, sua mão é composta de cartas totalmente ordenadas. Então, você recebe outra carta e repete o mesmo procedimento. Então outra carta, e outra, e assim por diante, até você não receber mais cartas. Esta é a ideia por trás da ordenação por inserção. Percorra as posições do array, começando com o índice 0 (primeiro elemento). Cada nova posição é como a nova carta que você recebeu, e você precisa inseri-la no lugar correto no subarray ordenado à esquerda daquela posição[2].

   Pois bem, declare um vetor de inteiros com 10 elementos e solicite ao usuário que digite os elementos, um por um, até completar o vetor. A seguir, ordene o vetor (sem fazer cópia do mesmo) usando o algoritmo e, por fim, exiba o vetor ordenado na tela.

   Requisitos: Anterior + Procedimentos.

3) Em Análise Combinatória, definimos a Combinação ***n*** elementos (total de elementos) ***r*** a ***r*** (tamanho do subconjunto), como:

$$C_r^n = \binom{n}{r} = \frac{n!}{r! \cdot (n-r)!}$$

   Observe que o operador fatorial aparece nessa expressão. Elabore um programa que solicita ***n*** e ***r*** ao usuário, calcula C_r^n e exibe o resultado na tela. No entanto, o cálculo do fatorial deve ser feito por um módulo separado. Ou seja, tome o número que se deseja calcular o fatorial e passe-o como parâmetro para um procedimento/função que calculará o fatorial do número. Essa função/procedimento deverá retornar o resultado para o programa principal, que se encarregará usá-lo para calcular a Combinação e exibir o resultado na tela. OBS: Atenção para o fatorial máximo numa arquitetura de 16 bits! (oi, *overflow*...)

Do Relatório

É necessária a confecção de um relatório escrito do trabalho, correspondendo a 40% da nota, conforme o tópico anterior. Apesar de haver certo grau de liberdade na elaboração do relatório escrito, os seguintes tópicos devem ser abordados:

- Todos os aspectos da linguagem de baixo nível que a implementação em C simplesmente "oculta", ou seja, você sequer precisa saber que aquilo existe em Assembly ao implementar em C. Dê exemplos.
- Comparação da quantidade de instruções relevantes presentes em C/Assembly;
- Explicação do funcionamento básico de todas os macros/procedimentos que foram copiados do inc\emu8086.inc.
- Avaliação do grupo do tempo necessário para cada implementação. Tome a implementação em C como referência e estime o percentual de tempo a mais necessário para a implementação em Assembly.
- Conclusão, incluindo uma auto avaliação do aprendizado adquirido;

---

[1] Ou Algoritmos e Estruturas de Dados, no caso dos alunos de Eng. de Controle e Automação

[2] Descrição disponível em https://pt.wikipedia.org/wiki/Insertion_sort

DA ENTREGA

A data limite para entrega do trabalho, via email do professor, é o dia **05/07 (Quinta-feira)** até as **23:59**.

Instruções para envio do email com o trabalho:

Assunto: **[AOC - 2018-1] Trabalho 2 - Aluno1, Aluno2, Aluno3, Aluno4, Aluno 5**

Anexo: Pacote compactado (zip/rar) com o mesmo nome usado no campo assunto e contendo os 6 arquivos mencionados anteriormente e um relatório em formato PDF conforme recomendações acima citadas (`relatorio.pdf`).

Siga estritamente as regras acima, ou seu trabalho poderá ser desclassificado. Principalmente as normas de nomenclatura. É SÉRIO! SEREI MUITO RÍGIDO QUANDO A ISSO E NÃO ACEITAREI RECLAMAÇÕES SOB HIPÓTESE ALGUMA! O objetivo é facilitar a correção/análise por parte do professor.

Bom Trabalho!