UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DE COMPUTAÇÃO

SCC0502 - Algoritmos e Estruturas de Dados I

Prof. Dr. Diego Raphael Amancio
diego@icmc.usp.br
Estagiária PAE: Vanessa Queiroz Marinho
vanessa.qm.1@gmail.com

Trabalho 1 - Minha primeira calculadora
Entrega no run.codes: 04/10

Especificação: Suponha uma máquina de calcular que trabalha apenas com números não negativos, e que tem apenas as quatro operações: soma, subtração, produto e divisão inteira. A máquina tem 16 teclas, representadas pelos caracteres:

0 1 2 3 4 5 6 7 8 9 + - * / C E

onde C representa "clear", e E representa "enter", que é usado indicar que vai ser fornecido um número. A máquina usa notação Polonesa Reversa, aquela em que o operador vem depois dos operandos.

Escreva um programa que usa uma pilha de inteiros para simular a máquina: cada caractere que entra é tratado, e a resposta é o conteúdo da pilha da máquina. Inicialmente, a pilha da máquina está vazia. As ações correspondentes a cada caractere são:

`i = 0, ⋯, 9`	troque o valor `x` do topo da pilha por `x*10+i`
`E`	empilhe um `0`
`op = +, -, *, /`	tire dois elementos `y` e `x` do topo da pilha, e empilhe `x op y`
`C`	esvazia a pilha

Por exemplo, se as entradas fossem sucessivamente

E 9 0 E 2 0 E 1 5 E 1 3 - * E 5 + /

a pilha teria sucessivamente os seguintes conteúdos:

(inicial)	[ ]
E	[0]
9	[9]
0	[90]
E	[90 0]
2	[90 2]
0	[90 20]
E	[90 20 0]
1	[90 20 1]
5	[90 20 15]
E	[90 20 15 0]
1	[90 20 15 1]
3	[90 20 15 13]
-	[90 20 2]
*	[90 40]
E	[90 40 0]
5	[90 40 5]
+	[90 45]

indicando o resultado 2 de 90 20 15 13 - * 5 + /, ou seja, (90/((20*(15-13))+5)) na notação habitual infixa.

Entrada

A entrada será composta por uma cadeia de caracteres como descrita anteriormente. Não é preciso verificar a corretude da entrada.

Exemplo

```
E 9 0 E 2 0 E 1 5 E 1 3 - * E 5 + /
```

Saída

Será composta do estado da pilha a cada passo, indicando finalmente o resultado.

Exemplo

```
-
0
9
90
90 0
90 2
90 20
90 20 0
90 20 1
90 20 15
90 20 15 0
90 20 15 1
90 20 15 13
90 20 2
90 40
90 40 0
90 40 5
90 45
2
```

A formatação da saída deverá respeitar as seguintes regras: a) a pilha vazia será representada por um hífen: - sem espaços antes ou depois do símbolo. b) cada número é seguido por um espaço e ao final da impressão do estado da pilha, o programa deverá realizar uma quebra de linha. Veja o exemplo abaixo, onde o caractere sublinhado _ representa um espaço e ¥n uma quebra de linha

```
90_20_15_0_¥n
```

ESTENDA a funcionalidade da calculadora incluindo:

a) um novo operador binário ^ como exponenciação (x ^ y), em que o resultado é x^y e

b) um unário ! como fatorial (n!), em que o resultado é n!

Para resolver o trabalho, o grupo deve implementar o TAD PILHA de forma ENCADEADA DINÂMICA (como visto em aula, "Aula 3: Pilhas - Parte 2"), com as operações disponibilizadas no arquivo ".h" e no ".c" do TAD PILHA.

Os trabalhos serão avaliados de acordo com os seguintes critérios:

1. **Corretude** do programa e do TAD: o programa deve fazer o que foi especificado;

2. **Casos de teste** definidos no run.codes
3. Estruturas de dados utilizadas: **adequação e eficiência;**
4. **Observação das boas práticas da programação.**

Os trabalhos devem ser implementados em C.

Observação sobre os grupos: Os grupos devem ser formados por no máximo 3 alunos, e estes grupos se manterão constantes ate o fim da disciplina.

Entrega: submissão de arquivo zip no run.codes até às 11:59 do dia **04/10.**

O que entregar?
Um arquivo zip com:

- arquivos de código-fonte do programa;
- arquivo Makefile;
- Relatório sucinto e objetivo informando os membros do grupo (número USP), contendo, pelo menos, (a) breve descrição do trabalho, com a estrutura de dados utilizada, (b) os casos de testes, seus comentários, telas de saída.

Exemplo de arquivo Makefile:

```
all:
	gcc -o my_app main.c foo.c
clean:
	rm my_app
run:
	./my_app
```

Prazo de entrega: **04/10** até às 11:59. A cada dia de atraso, um ponto a menos. Se cópia identificada, zero para todos os grupos envolvidos.