Data de Entrega: NO SIGAA – NÃO SERÃO ACEITOS TRABALHOS ENVIADOS APOS ESTA DATA.
O que deve ser entregue: 1. Deve ser enviado por email um arquivo zipado contendo os códigos fontes dos programas implementados e os documentos solicitados. Use o nome especificado em cada questão para nomear o seu programa;
Regras para entrega do trabalho prático: Este trabalho deve ser realizado em grupo de no máximo 3 alunos. Cada grupo deve escolher um representante que será responsável por enviar o trabalho em um arquivo zipado, cujo nome deve respeitar o seguinte formato: CK069_<matricula do aluno>_T<numero do trabalho pratico>, por exemplo, CK069_123456_T1 é o arquivo contendo o trabalho pratico 1 do estudante com matricula 123456. O arquivo compactado deve ser enviado para o SIGAA até meia-noite do prazo estipulado acima. Caso o aluno não respeite as regras apresentadas anteriormente, poderá sofrer redução na nota do trabalho.
Como será avaliado o trabalho: A avaliação deste trabalho levará em conta a correção e qualidade das questões respondidas. Cada questão descrita abaixo possui uma pontuação definida. Este trabalho deve ser apresentado ao monitor no dia a ser especificado pelo professor

Trabalho Prático 6 – Algoritmos Safety, Avoid e Detection

Neste laboratório você deve implementar 3 algoritmos relacionados com o gerenciamento de deadlock. Os algoritmos que devem ser implementados devem ser; Safety (verificar se o sistema está em estado seguro); Avoid (no caso de uma solicitação de recurso por um processo) e Detection (verificar se sistema entrou em deadlock). Para implementar esses algoritmos, você deve receber como entrada um conjunto de processos, uma quantidade de tipos de recursos, as necessidades de recursos dos processos e um estado inicial de solicitação de recursos pelos processos:

1. (3,0 pontos) implemente o algoritmo *Safety* para verificar se existe uma sequência de execuções dos processos segura. Utilize como dados para a execução de seu algoritmo o seguinte exemplo. Crie outros exemplos para validar seu algoritmo.

	Alocado no momento t0			Máximo requerido pelo processo		
	A	B	C	A	B	C
P0	3	1	0	7	5	7
P1	7	0	1	7	1	1
P2	0	2	2	10	4	3

2. (3,0 pontos) Dada uma nova solicitação de recurso para um processo, implemente o algoritmo *Avoid* para verificar se é possível permitir esta solicitação deste recurso, evitando um deadlock. Caso seja detectada a possibilidade do sistema entrar em deadlock, então o processo deverá ser bloqueado.

3. (2,5 pontos) Implemente o algoritmo *Detection* para detectar a existência de deadlock em um conjunto de processos em execução. O algoritmo deve imprimir o numero do processo em deadlock. A partir do exemplo de dados apresentado acima crie um conjunto de dados que gere um deadlock.