Nome: Luiz Gabriel Zeferino Duarte

RA: n454cd-8

1) Um processo é um programa único em execução, ou seja, uma thread de execução, que consome recursos computacionais para funcionar. Processos paralelos são aqueles que não disputam a mesma área de recurso, portanto podem ser executados paralelamente (um não interfere no outro). Já processos concorrentes são o oposto, ambos precisam consumir uma mesma área de recursos, que ganha o nome de zona crítica, portanto, um processo pode prejudicar a execução do outro se não forem bem sincronizados .

2)

```
boolean ponte_livre = true;  /* esse é o semáforo */

char sentido_ponte;  /* essa variável assume o valor "n" se o sentido da ponte estiver norte-sul, e "s" se tiver sul-norte */

void carro_norte(){
        while (!ponte_livre){              // estratégia parecisa com busy waiting
                if (sentido_ponte == n)passe();
                else espere();
        }
        ponte_livre = false;
        sentido_ponte = n;
        passe();
        ponte_livre = true;
}
```

3) A diferença primordial entre busy waiting e blocking é que em busy waiting o processo continua executando, porém gasta esse tempo apenas para chegar se uma zona crítica foi liberada, enquanto no bloking, o processo sai do estado de execução e fica parado esperando um retorno, o que não gasta clocks da cpu.

4)

```
boolean semaforo_mutex = true;

int garfos = 2;

void pensar(){

}

void comer(){

}

void pegar_garfo (){

        while (!semaforo_mutex){}

                semaforo_mutex  = false; /*deixar o semáforo em false para nenhum processo acessar a zona crítica*/

                if( garfo >= 1){

                        garfo = garfo - 1;

                }

                semaforo_mutex  = true; /*liberar a zona crítica*/

}

void soltar_garfo(){

                while (!semaforo_mutex){}

                semaforo_mutex  = false; /*deixar o semáforo em false para nenhum processo acessar a zona crítica*/

                        garfo = garfo + 1;

                        semaforo_mutex  = true; /* liberar a zona crítica*/
```

```
}
void filósofo ()
{
  while (true) {
    pensar();
    pegar_garfo();
    comer();
    soltar_garfo();
  }
}
```

5) Os requisitos mínimos para se implementar uma memória virtual são possuir espaço no hd suficiente para isso e houver muitos elementos em blocking ocupando grandes espaços de memória, assim, a fim de não deixar o processador ocioso, esses processos bloqueados sofrem swap e vão para a memória virtual, onde ficam suspensos. Assim liberando espaço na memória para mais processos.