Parte A (10 valores)

1. Os sistemas de operação de uso geral actuais são tipicamente *sistemas de operação de rede*. Faça a sua caracterização.

2. O que são semáforos? Mostre como é que eles podem ser usados para garantir acesso a uma região crítica com exclusão mútua.

3. Distinga *deadlock* de *adiamento indefinido*. Qual é a diferença entre políticas de prevenção de *deadlock* no sentido estrito e no sentido lato. Dê um exemplo de cada uma delas.

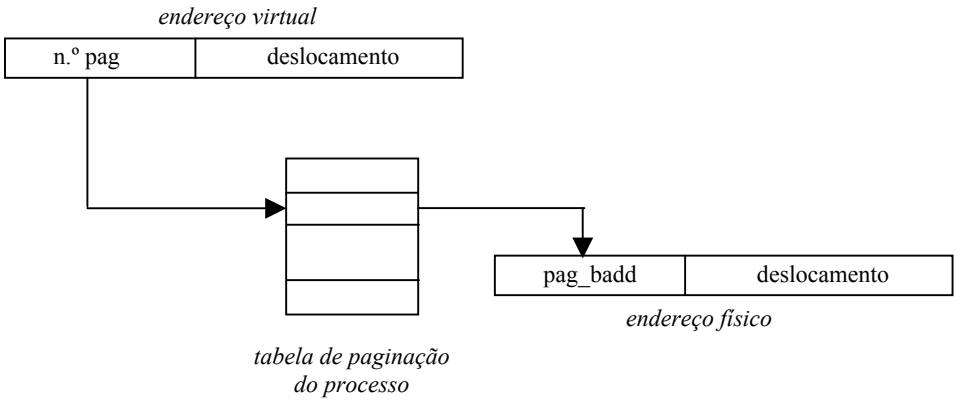
4. O que distingue dispositivos de tipo caracter de dispositivos de tipo bloco? Descreva de uma maneira funcional como se desenvolve a comunicação entre um processo utilizador e um dispositivo de tipo caracter.

5. Em que consiste o mecanismo de *spooling*? Mostre como é que ele pode ser utilizado no caso da impressora. Podem evitar-se sempre situações de deadlock? Como? Qual é o seu inconveniente?

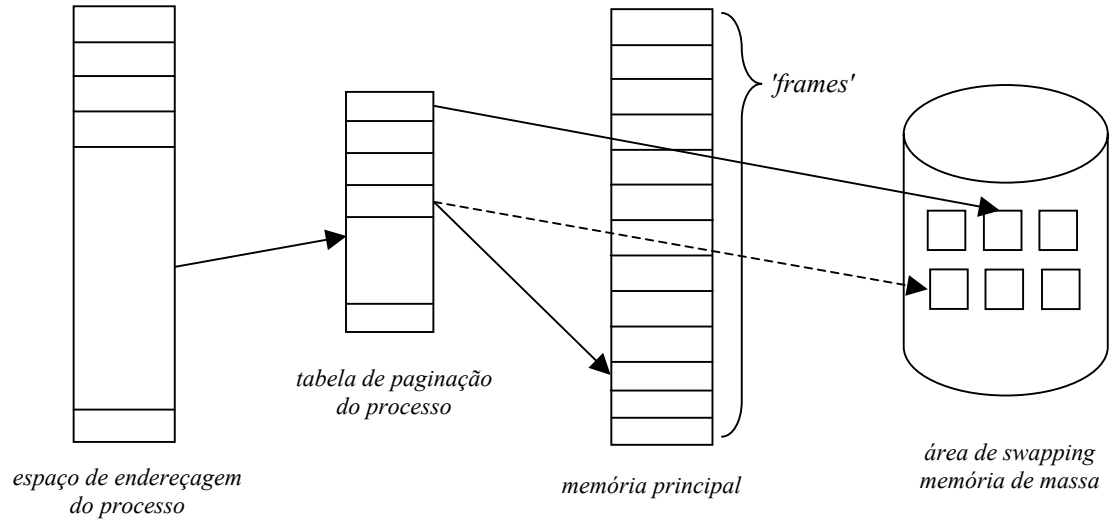
Parte B (10 valores)

Considere uma organização de memória virtual paginada em que as páginas têm o tamanho de 4KB.

A figura abaixo representa de uma maneira simplificada o mecanismo de tradução de um endereço virtual num endereço físico. Assuma endereços de 32 bits (unsigned long).



A figura seguinte procura descrever a distribuição das páginas do espaço de endereçagem de um processo num dado instante, onde parte delas está localizada em memória principal.



O algoritmo de substituição de páginas utilizado é o NRU (*Not Recently Used*).

Admita que foram definidas as estruturas de dados seguintes:

Entrada (simplificada) da Tabela de Controlo de Processos

```
typedef struct
        { BOOLEAN busy;                 /* sinalização de entrada ocupada */
          unsigned int pid,                  /* identificador do processo */
                       pstat;              /* estado do processo: 0 - RUN;
                                             1 - BLOCKED; 2 - READY-TO-RUN
                                               3 - CREATED; 4 - TERMINATED */
          unsigned char intreg[K];            /* contexto do processador */
          unsigned long pag_tb;       /* endereço da tabela de paginação
                                         do processo em memória principal */
          unsigned int pag_tb_len;         /* n.º de entradas da tabela */
        } PCT_ENTRY;
```

Entrada (simplificada) da Tabela de Paginação

```
typedef struct
        { BOOLEAN loaded;     /* sinalização de carregamento da página em
                                                          memória principal */
          unsigned long pag_badd,   /* localização do início da região de
                              armazenamento da página em memória principal */
                        pag_blk;      /* n.º do bloco de armazenamento da
                                             página na área de swapping */
          BOOLEAN acss,              /* sinalização de acesso à página no
                                        último intervalo de monitorização */
                  modif;          /* sinalização de modificação da página
                               desde que foi carregada em memória principal */
        } TB_PAG;
```

Nó de lista biligada

```
struct binode
      { unsigned int pct_index;           /* índice da entrada da PCT que
                                   descreve o processo a que pertence o nó */
        unsigned int npag;    /* n.º da página do espaço de endereçamento */
        unsigned int nobjt;  /* n.º do frame ou do bloco de armazenamento */
        struct binode *ant,                /* ponteiro para o nó anterior */
                      *next;               /* ponteiro para o nó seguinte */
      };
typedef struct binode BINODE;
```

CAM

```
struct cam
      { BINODE *pstart;               /* ponteiro para o início da CAM */
        int n;                                    /* tamanho da CAM */
      };
typedef struct cam CAM;
```

e as variáveis globais descritas abaixo:

```
static PCT_ENTRY pct[100];            /* tabela de controlo de processos */
static unsigned int pindex;              /* índice da entrada da PCT que
                                descreve o processo que detém o processador */
static CAM b_frm,      /* lista dos frames ocupados da memória principal */
           f_frm;        /* lista dos frames livres da memória principal */
static CAM b_blk,       /* lista dos blocos ocupados da área de swapping */
           f_blk;         /* lista dos blocos livres da área de swapping */
```

Finalmente, as primitivas seguintes estão também disponíveis:

Salvaguarda e restauro do contexto

```
void save_context (unsigned int pct_index);
void restore_context (unsigned int pct_index);
```

Reserva e libertação de espaço em memória dinâmica

```
void *malloc (unsigned int size);
void free (void *pnt);
```

Inserção e retirada e pesquisa de nós na CAM (os nós são ordenados pelo campo *nobjt*)

```
void cam_in (unsigned int nord, CAM *cam, BINODE *val);
void cam_out (unsigned int nord, CAM *cam, BINODE **val_p);
          se o nó não existir, *val_p = NULL
BOOLEAN cam_empty (CAM *cam);
void cam_search (unsigned int nord, CAM *cam, BINODE **val_p);
          se o nó não existir, *val_p = NULL
```

Transferência de páginas de e para a memória principal

```
void swap_in (unsigned int npag, unsigned int nblk);
void swap_out (unsigned int npag, unsigned int nblk);
```

1. Descreva o conteúdo da tabela de paginação de um processo cujo espaço de endereçagem é formado por 4 páginas com as características seguintes:

   - o seu armazenamento na área de 'swapping' é feito, respectivamente, nos blocos n.º $1023D_{16}$, $F54A_{16}$, 25_{16} e $5C567_{16}$;
   - a primeira e a terceira páginas estão residentes em memória principal nos 'frames' 195_{16} e $5A_{16}$;
   - a terceira página foi modificada e ambas foram referenciadas no último último intervalo monitorado.

2. Caso ocorra um 'page fault', descreva sumariamente as operações que devem ser efectuadas pela rotina de serviço à interrupção. Em que estado deve ser colocado o processo?

3. Construa a primitiva que retira o processo de memória, faz o seu 'swapped out' completo após ele ter terminado. Não se esqueça de actualizar os campos da respectiva entrada da tabela de controlo de processos.

   ```
   void full_swap_out (unsigned int pct_index);
   ```

4. Construa a primitiva de substituição do conteúdo de um 'frame' em memória principal de acordo com o algoritmo NRU. Assuma que a substituição se fará entre um dos 'frames' atribuídos ao processo.

   ```
   void NRU_replace (unsigned int npag);
   ```