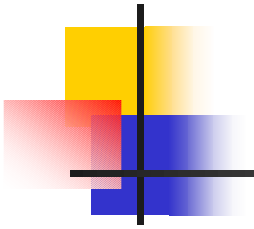
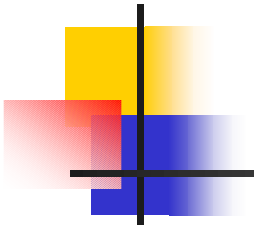


# Trab. 13

- Considere um cadastro de alunos armazenado numa tabela de dispersão (*hash*) que usa como chave de busca o número de matrícula. Considere também a existência de uma função de *hash* que, dado o número de matrícula, retorna o índice na tabela. O protótipo dessa função é dado por:

  int hash (int mat);

- O tratamento de colisão é feito utilizando-se listas encadeadas para armazenar os elementos que colidem. Dessa forma, cada posição na tabela representa um ponteiro para o primeiro nó de uma lista encadeada.

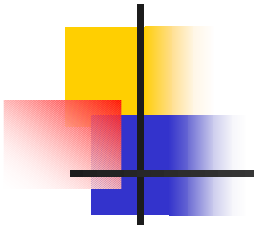


# Trab. 13

- Considere as seguintes declarações, relativas à tabela de *hash* dos alunos.

```
struct aluno{
    int matric;              //matrícula do aluno
    char nome[51];           //nome do aluno
    struct aluno* aluno prox;  //prox. elem. da lista
};
typedef struct aluno Aluno;

#define N 127
typedef Aluno* Hash[N];
```



# Trab. 13

- Considerando que posições não preenchidas da tabela de *hash* armazenam o valor NULL (lista vazia), escreva uma função de busca que verifica se um aluno, dado seu número de matrícula, está armazenado no cadastro. A função deve retornar o ponteiro da estrutura Aluno se estiver no cadastro ou NULL se não estiver.

- O protótipo da função deve ser

  Aluno * busca (Hash tab, int mat);