Universidade Federal do ABC
Algoritmos e Estruturas de Dados II 3^o Quadrimestre 2018

Prof. Carlos da Silva dos Santos

Aula prática 02 – Funções de Espalhamento (*hashing*).

1 - Método da divisão: no método da divisão, a função de espalhamento é definida como

$$h(k) = k \mod m \tag{1}$$

em que k é um número natural (chave), $h(k)$ é a função de espalhamento e m é um número natural. Os valores de $h(k)$ ficam então restritos ao intervalo $\{0, 1, 2, \ldots, (m-1)\}$. Em geral, recomenda-se que m seja escolhido como um número primo, não muito próximo de alguma potência de 2. Implemente uma função `div_hash` para calcular $h(k)$, seguindo o protótipo abaixo:

```
int div_hash(int key,  int m);
```

em que `key` é a chave e `m` é o parâmetro da fórmula acima.

(a) Usando $m = 12$, faça um programa que teste sua função com valores de chave variando de 0 até 100. Quando o resultado $h(x)$ for igual a 3, imprima o valor de chave correspondente. Você consegue notar um padrão para esses valores de chave?

(b) Repita o item anterior com $m = 11$. Você consegue notar um padrão para esses valores de chave?

(c) Usando $m = 97$ (um número primo) conte o número de colisões para cada valor diferente de $h(k)$, usando chaves no intervalo $\{1, 2, 3, \ldots, 10000\}$. *Dica*: você pode acumular as contagens em um vetor de m posições, inicialmente preenchido com zeros. A cada vez que você calcular um novo valor $h(k)$, incremente a posição correspondente no vetor de contagens. Salve os resultados dessas contagens em um arquivo e faça um gráfico de *número de colisões* em função do valor do *hash*. Você pode salvar as contagens em um arquivo de *valores separados por virgula*, em que cada linha tem o formato `chave, contagem`. O gráfico pode ser construído em um programa qualquer de planilhas, por exemplo.

2 - Método da multiplicação: no método da multiplicação, a função de espalhamento é definida como

$$h(k) = \lfloor m \times ((k \times A) \mod 1) \rfloor \tag{2}$$

Em que k é a chave, m é o número de posições da tabela, A é uma constante não negativa, $0 < A < 1$. O símbolo $\lfloor x \rfloor$ representa a função *chão*, isto é, o maior inteiro que seja menor que x (arredondamento de x "para baixo"). A notação
Implemente uma função `mul_hash` para calcular $h(k)$, seguindo o protótipo abaixo:

```
int mul_hash(int key,  int m, float a);
```

em que `key` é a chave, `m` e `a` correspondem aos parâmetros da fórmula acima.
Dica: Para usar a função `floor` (chão) em `C`, você precisar incluir o arquivo de cabeçalho `math.h`. As funções matemáticas estão reunidas em uma biblioteca chamada `libm.a` (estática) ou `libm.so` (dinâmica). Para compilar o seu programa, será então necessário *linkar* com a biblioteca correspondente, adicionando o parâmetro `-lm` *ao final* do comando de compilação. Por exemplo, caso o seu programa esteja no arquivo `fonte.c`, use o comando:

```
gcc -o programa -Wall fonte.c -lm
```

(a) Usando $m = 200$ e $A = 0.62$, faça um programa que teste sua função com valores de chave variando de 1 até 500 mil. Conte o número de colisões para cada valor diferente de $h(k)$. Salve os resultados dessas contagens em um arquivo e faça um gráfico de *número de colisões* em função do valor do *hash*.

(b) Usando $m = 200$ e $A = 0.61803398875$ (número derivado da razão áurea). Compare os resultados de distribuição das colisões.