Programação Imperativa 2021/2022 (CC1003), DCC/FCUP

Folha 8

8.1 Sejam `i` uma variável inteira e `p` e `q` apontadores para inteiros. Indique quais das seguintes atribuições são legais.

1. `p = i;`
2. `*p = i;`
3. `p = &q;`
4. `p = &i;`
5. `&i = p;`
6. `p = q;`
7. `p = *q;`
8. `*p = q;`
9. `*p = *q;`

8.2 Sejam `i` uma variável `int` e `p` um apontador para `i`. Indique quais das seguintes instruções de leitura e escrita são corretas.

1. `printf("%d",i);`
2. `scanf("%d",i);`
3. `scanf("%d",&i);`
4. `printf("%d",p);`
5. `printf("%d",*p);`
6. `scanf("%d",p);`
7. `printf("%p",p);`
8. `printf("%p",i);`
9. `printf("%p",*p);`

8.3 Escreva uma função

```
void decompor(int total_seg, int *horas, int *mins, int *segs);
```

que decompõe um total inteiro de segundos `total_seg` em horas, minutos (0-59) e segundos (0-59); os resultados devem ser atribuídos ao conteúdo dos apontadores `horas`, `mins` e `segs`. Pode assumir que o total de segundos é maior que zero.

8.4 Escreva uma função

```
void max_min(int vec[], int size, int *pmax, int *pmin);
```

que determina o valor máximo e mínimo de um vetor; os resultados devem ser atribuídos ao conteúdo dos apontadores `pmax` e `pmin`. Pode assumir que `size` é sempre maior que zero.

8.5 Escreva uma função `void reduzir(int *pnum, int *pdenom)` que reduz uma fração de numerador e denominador `*pnum` e `*pdenom` à forma simplificada. O numerador e denominador devem ser modificados por meio dos apontadores `pnum` e `pdenom`. Pode assumir que o denominador é sempre diferente de zero.

8.6 Re-escreva a função seguinte para usar apontadores em vez de índices (ou seja: eliminar a variável `i` e os usos de indexação `vec[...]`).

```
void store_zeros(int vec[], int n){
   int i;
   for(i=0; i<n; i++)
      vec[i]=0;
}
```

8.7 Considere a função apresentada na aula teórica 9 para contar espaços de uma cadeia de carateres. Pretende-se que escreva uma versão alternativa desta função usando apontadores em vez de índices. A função dever ter a declaração

```
int contar_espacos(char *str);
```

O resultado deve ser o número de espaços na cadeia.

8.8 Considere a função apresentada na aula teórica 9 para inverter a ordem de carateres de uma cadeia. Pretende-se que escreva uma versão alternativa desta função usando apontadores em vez de índices. A função dever ter a declaração `void inverter(char *str);`

8.9 (**Plataforma codex**) Considere a função apresentada na aula teórica 9 para procurar um carater numa cadeia. Pretende-se que escreva uma versão alternativa desta função usando apontadores em vez de índices. A função dever ter a declaração

```
char *procurar(char *str, char ch);
```

O resultado deve ser um apontador para a primeira ocorrência do carater `ch` (se este ocorrer) ou `NULL` caso contrário.

8.10 Considere a função apresentada na aula teórica 9 para comparar igualdade de cadeias de carateres. Pretende-se que escreva uma versão alternativa desta função usando apontadores em vez de índices. A função dever ter a declaração

```
int comparar(char *str1, char *str2);
```

O resultado deve ser 1 se as cadeias são iguais e 0 se são diferentes.