Estruturas de Dados I

Apontadores – revisão (ficha 1)

1. O programa seguinte tem como objetivo imprimir os valores de `var1` e `var2` e os seus endereços (guardados em `ptr1` e `ptr2`).

```
#include <stdio.h>
int main(){
    int var1=5;
    char var2='a';
    int *ptr1= &var1;
    char *ptr2;
    *ptr2='b';
    printf("var1 tem o endereço %p e o valor %d\n", ptr1, var1 );
    printf("var2 tem o endereço %p e o valor %c\n", ptr2, *ptr2 );
}
```

   Siga os seguintes passos e interprete, para cada caso, os valores apresentados pelo programa:

   (a) Se tentar compilar e executar este programa ocorrerá o erro `Segmentation fault`. Corrija o programa para que isso não aconteça.

   (b) Altere o programa para que este imprima também o tamanho dos tipos de variáveis `char`, `char *`, `int` e `int *`.

   (c) Altere o programa para que este imprima ainda os valores de `prt1+1` e `ptr2+1` e compare com os valores de `ptr1` e `ptr2`.

2. Considere o programa seguinte e complete as tabelas.

```
#include <stdio.h>
int main(){
    int i, j, *p1, *p2, **pp1, **pp2;
    i=4;
    j=5;
    p1 = &i;
    p2 = &j;
    pp1 = &p2;
    pp2 = &p1;
}
```

variavel	i	j	p1	p2	pp1	pp2
conteudo	4	5				
endereço	1000	1007	1030	1053	1071	1079

expressão	i	*p2	&i	&p2	*pp1	*pp2	&(*p1)	j	*p1	*(&p1)
conteudo										

3. Escreva a função `void ordena( int *a, int *b, int *c )` que recebe 3 valores e ordena-os, colocando o menor em `a` e o maior em `c`. Teste a função com um programa que pede ao utilizador 3 números, invoca a função `ordena` e imprime o resultado no ecrã.

```
Insira os valores a ordenar: 43 65 17
Valores a, b, c ordenados por ordem crescente: 17 43 65
```

4. Escreva uma função que recebe dois apontadores para inteiros (passagem de parâmetros por referência) e devolve o endereço do maior valor. Escreva um programa para testar a função. Considere o protótipo `int *vmaior( int *v1, int *v2 )`.

```
Insira dois valores: 43 65
Maior: endereco=0016F838, valor=65
```

5. Escreva a função `int horasMin( int min, int *horas, int +minutos )` que converte um valor em minutos para horas e minutos. A função devolve 1 se o o tempo em `min` for superior a um dia e 0 caso contrário. Escreva um programa para testar a função que pede ao utilizador um inteiro, invoca a função `horasMin` e imprime o resultado no ecrã.

```
Insira o total de minutos: 568
568 minutos correspondem a 09h:28min; não é superior a um dia.
```

```
Insira o total de minutos: 4689
568 minutos correspondem a 78h:09min; é superior a um dia.
```

6. Escreva uma função que faça a conversão entre o espaço de cor RGB e o espaço de cor HSV. A conversão entre espaços de cor é efetuada da seguinte forma:

   1. Converter os valores RGB que estão na gama [0..255] para a gama [0..1]
   2. Encontrar o máximo e o mínimo das componentes RGB
   3. Aplicar as seguintes equações:

$$H = \begin{cases} 60 \times \frac{G-B}{max-min}, \text{ se } max = R \text{ e } G \geq R \\ 60 \times \frac{G-B}{max-min} + 360, \text{ se } max = R \text{ e } G < R \\ 60 \times \frac{B-R}{max-min} + 120, \text{ se } max = B \\ 60 \times \frac{R-G}{max-min}, \text{ se } max = B \end{cases}$$

$$S = \frac{max - min}{max}$$

$$V = max$$

   Nota: R,G,B∈[0..255], H∈[0,360], S∈[0,1], V∈[0,1].

   Escreva um programa para testar a função. Assuma o seguinte protótipo para a função `void rgb2hsv( int R, int, G, int B, float *h, float *s, float *v )`

```
Insira as componenete RGB: 123 234 80
RGB=(123, 234, 80) <=> HSV=(103.247, 0.658, 0.918).
```