Universidade Estadual Vale do Acaraú
Curso de Ciências da Computação
Disciplina de Laboratório de Programação
Pf. Paulo Regis Menezes Sousa

Avaliação Parcial 2

Questão 1. 1 P.

Crie uma função que recebe uma string e um caractere, e apague todas as ocorrências desses caractere na string substituindo cada ocorrência por um espaço em branco.

Questão 2. 1 P.

Crie uma função que recebe uma string de letras minúsculas e transforma alguns dos caracteres em maiúsculos. Faça sorteios com a função `rand` para escolher os índices dos caracteres que serão alterados.

Questão 3. 1 P.

0.5P Crie uma estrutura que represente um super-herói, a estrutura deve ter os membros: nome (*string*), força, resistência e velocidade (*float*).

0.5P Pergunte ao usuário o número de heróis que deseja criar, aloque dinamicamente um vetor do tipo `struct heroi` e ao final da inserção dos heróis, imprima o vetor mostrando as informações de cada herói.

Questão 4. 2 P.

1.0P Crie uma função que receba como parâmetros duas variáveis do tipo `struct heroi`. A função deve comparar e alterar os membros de cada estrutura da seguinte forma:

1. O campo **velocidade** do primeiro é comparado ao campo **velocidade** do segundo.
   (a) Se $velocidade_1 > velocidade_2$ então você deve diminuir a *resistência* do segundo em 0.5 pontos.
   (b) Se $velocidade_1 < velocidade_2$ então você deve diminuir a força do primeiro em 0.5 pontos.
   (c) Se $velocidade_1 == velocidade_2$ não faça nada.
2. O campo **força** do primeiro é comparado ao campo **resistência** do segundo.
   (a) Se $força_1 > resistência_2$ então você deve diminuir a *resistência* do segundo em 2.5 pontos. A função então retorna 1 indicando que o primeiro venceu.
   (b) Se $força_1 < resistência_2$ então você deve diminuir a força do primeiro em 0.5 pontos. A função então retorna 2 indicando que o segundo venceu.
   (c) Se $força_1 == resistência_2$ a função então retorna 0 indicando que houve empate.

1.0P Na função `main` use a função de comparação, crie dois exemplos mostrando as comparações entre heróis diferentes, exiba o nome dos vencedores de cada comparação e mostre os valores dos membros de cada estrutura com os valores alterados após o uso da função.

Questão 5. 2 P.

1.0P Implemente a função `imprimir_vetor` para imprimir as strings do vetor `vet_str[]` do código abaixo.

1.0P Implemente a função `comparador` para que as strings sejam ordenadas de forma decrescente segundo a ordem alfabética.

```c
#include <stdio.h>
#include <stdlib.h>
#define TAM 5

void imprimir_vetor(char *v[], int n);
int comparador(const void *a, const void *b);

int main() {

     char *vet_str[TAM] = {"abacaxi","cenoura","banana","beterraba","goiaba"};

     imprimir_vetor(vet_str, TAM);
     qsort(vet_str, TAM, sizeof(char*), comparador);
     imprimir_vetor(vet_str, TAM);

     return 0;

}
```

Questão 6. 3 P.

2P Crie uma função `map` com a seguinte assinatura:

```c
float *map(float v[], int l, float k, float (*fun)(float a, float b));
```

onde, `v[]` é um vetor de números reais, `l` é o tamanho do vetor, `k` é um valor real e `fun` um ponteiro para função.

A função `map` deve alocar dinamicamente um vetor de mesmo tipo e tamanho de `v[]`. Cada posição do novo vetor `new_v[]` será preenchida com o resultado da função `fun` aplicada à respectiva posição de `v[]` e ao valor `k`: `new_v[i] = fun(v[i], k)`. Após percorrer todo o vetor `map` retorna um ponteiro para o novo vetor `new_v[]`.

1.0P Na função `main` peça ao usuário um número inteiro *N*. Aloque dinamicamente um vetor `v[]` *float* de tamanho *N*. Peça ao usuário que preencha `v[]`, aplique ao vetor a função `map` para criar um novo vetor `new_v[]` e imprima o novo vetor.