PUC-Rio – Programação 2 – INF1007
Prova 3 – Soluções – Turma 33E

- Questao 1: (Félix Ribeiro)

```
void decifra (char * cod)
{
        int i;
        PilhaChar * p = pilha_cria();

        /* Percorre a string até o final */
        for(i = 0 ; cod[i] != '\0' ; i++){

                /* Enquanto o caracter atual for uma consoante
                   vai empilhando e contiua percorrendo : por isso o i++ */
                while(cod[i] != 'A' && cod[i] != 'E' &&
                      cod[i] != 'I' && cod[i] != 'O' &&
                      cod[i] != 'U'  )
                {
                        pilha_push(p, cod[i]);
                        i++; /* para continuar percorrendo a string */
                }

                /* Quando sai do while de cima, chegou numa vogal.
                   Então temos q desempilhar as consoantes e imprimi-las */
                while(!pilha_vazia(p))
                        printf("%c",pilha_pop(p) );

                /* Depois de imprimir as consoantes empilhadas,
                        temos que imprimir a vogal corrente, pois o i
                        parou numa vogal no primeiro while */
                printf("%c", cod[i] );

        }

        pilha_libera(p);
}
```

- Questão 2 : (Guilherme Berger)

```
void exibeNomes(Noarv *a. int x, char nome[]) {
        if (a == NULL) return;

        if (a->info->inscricao <= x) {

                if (strcmp(a->info->nome, nome) != 0)
                        printf("%s\n", a->info->nome);

                exibeNomes(a->esq, x, nome);

                if (a->info->inscricao != x) /* considerando inscricao unica */
                        exibeNomes(a->dir, x, nome);
        }
        else
                exibeNomes(a->esq, x, nome);
}
```

- Questão 3 : (Daniel)

```
NoLista *eliminaNos(NoLista *l1, NoLista *l2) {
        NoLista *temp1 = l1;
        NoLista *temp2 = l2;
        NoLista *ant1 = NULL;

        while (temp1 != NULL && temp2 != NULL) {
                if (temp1->codigo == temp2->codigo) {

                        if (ant1 == NULL) /* eliminar o primeiro da lista 1? */
                                l1 = temp1->prox;
                        else
                                ant1->prox = temp1->prox;
                                temp1 = temp1->prox; /* avança nas duas listas */
                                temp2 = temp2->prox;
                }
                else if (temp1->codigo > temp2->codigo) /* avança apenas a segunda lista */
                        temp2 = temp2->prox;
                else {
                        ant1 = temp1; /* avança a primeira lista */
                        temp1 = temp1->prox;
                }

        }

        return l1;
}
```