PUC-Rio – Software Básico – INF1018
Prova 1 – 26/06/12 – Turma 3WA

1. (2,5 pontos) Considere o programa C a seguir:

```
#include <stdio.h>
void dump (void *p, int n) {
  unsigned char *p1 = (unsigned char *) p;
  while (n--) {
    printf("%p - %02x\n", p1, *p1);
    p1++;
  }
}
struct sno {
  float fval1;
  short sval;
  float fval2;
  double dval;
};
int main (void) {
  struct sno no = {-0.125, 1025, 2050.75, 65550};
  dump ((void *)&no, sizeof(struct sno));
  return 0;
}
```

   Considerando que `no` será alocado na posição de memória 0xcdfff7b0, diga o que esse programa irá exibir quando executado, explicando como você chegou aos valores exibidos (ATENÇÃO: valores sem contas e explicações NÃO valem ponto!!!). Suponha que a máquina de execução é *little-endian* e que as convenções de alinhamento são as do Linux no IA-32. Se houver posições de *padding*, indique seu conteúdo com `PP`.

2. (3,0 pontos) Traduza a função `foo` abaixo para assembly IA-32 (o assembly visto em sala), utilizando as regras usuais de alinhamento, passagem de parâmetros e resultados em C. Traduza o mais diretamente possível o código de C para assembly. Comente seu código.

   (Não se preocupe se você não entender o que a funçção faz, apenas traduza-a literalmente!)

   (a)
```
float aaa (int a, double b);

void foo (float *a, float *b, int n){
  int i;
  float acc = 0.0;
  int tam;
  for (i=0;i<n;i++)
    acc += aaa(i, a[i]);
  tam = acc;
  for (i=0;i<tam;i++)
    b[i] = acc;
}
```

3. Considere o programa formado pelos dois arquivos abaixo:

   - arquivo `t1.c`

```
#include <stdio.h>

extern char boo;
int max = 3;
```

```
void p2(int n) {
  int i;
  for (i=0;i<n;i++)
    printf ("%x\n", boo);
}
```

- arquivo `t2.c`

```
void p2 (int);
extern int max;

int boo () {
  return 1;
}

int main () {
  p2(max);
  return 0;
}
```

(a) (1,0 ponto) Suponha que compilemos esses dois arquivos em separado, gerando arquivos `t1.o` e `t2.o`. Liste todos os símbolos exportados (definidos pelo módulo) e importados (esperados de outros módulos) para cada um dos arquivos `t1.o` e `t2.o`, ou seja, o que apareceria como D (dados exportados) ou T (texto, ou código exportado), na primeira categoria, e, na segunda categoria, o que apareceria como U (undefined), na listagem do `nm`.

(b) (1,0 ponto) Ao executar o programa gerado a partir desses dois módulos, cada chamada a `printf` mostra a string `"0x55\n"`, e em seguida o programa termina normalmente. Explique o por que disso e diga o que você acha que é o valor `0x55`.

4. Dado o seguinte código de uma função na linguagem `SB` do segundo trabalho:

```
v0 = $0 + $0
if p1 3 7 3
if p0 7 7 4
v0 = v0 + p1
p0 = p0 - $1
if $1 3 3 3
ret v0
```

(a) (1,5 pontos) Escreva o código *assembly* da função, como seria criado pelo sua função *compila*. Use *assembly* e use labels simbólicos para os desvios.

(b) (1,0 ponto) Diga qual seria o resultado da chamada f(4,3).