PUC-Rio – Software Básico – INF1018
Prova 2 – Turma 3wb – 26/06/2018

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

```
#include <stdio.h>
void dump (void *p, int n) {
  unsigned char *p1 = p;
  while (n--) {
    printf("%p - %02x\n", p1, *p1);
    p1++;
  }
}
struct X {
  int i;
  double d;
  float f;
} x;
int main(void) {
  x.i = -2050;
  x.f = 18.25;
  x.d = -x.f;
  dump(&x, sizeof(x));
  return 0;
}
```

   Sabendo que o endereço do vetor `x` na memória é `0x601050`, e supondo que a máquina de execução é *little-endian* com as convenções de alinhamento do Linux no IA-64 (vistas em sala), **mostre o que esse programa irá imprimir quando executado**. Coloque **PP** nas posições correspondentes a *padding*.

   (Mostre como você chegou aos valores exibidos. Valores sem contas **NÃO** valem ponto!).

2. Traduza as funções `foo` e `boo` abaixo para assembly IA-64, utilizando as regras usuais de alinhamento, passagem de parâmetros, salvamento de registradores e resultados em C/Linux. (Não se preocupe em entender o que as funções fazem, apenas traduza-as literalmente.)

   Atenção! **Traduza o mais diretamente possível o código de C para assembly**.

   (a) (2,5 pontos)

```
int f(int x);
int g(int *p, int n);

int foo(int n) {
  int local[2] = {0,0};
  int i;
  while (n--) {
    i = f(n) & 1;
    local[i]++;
  }
  return g(local, 2);
}
```

(b) (2,5 pontos)

```
double f(double x);
double g(double x);

void boo(float *v, int n) {
  int i;
  for (i = 0; i < n; i++, v++) {
    *v = f(*v) / g(*v);
  }
}
```

3. (1,5 pontos) Considere os arquivos `arq1.s` e `arq2.s` a seguir:

arq1.s:

```
.data
.global v1
v1: .int 10
.text
.global foo
foo:
  cmpl $0, %edi
  jge  L1
  movl v1, %edi
L1:
  movl %edi, %eax
  ret
```

arq2.s:

```
.data
.global v1, v2
S1: .string "%d %d %d\n"
v1: .int 10
v2: .int 20
.text
.global main
main:
  movl $1, %edi
  call foo
  movl %eax, %esi
  movl v1, %edx
  movl v2, %ecx
  movq $S1, %rdi
  call printf
  movl $0, %eax
  ret
```

(a) Suponha que esses dois arquivos sejam processados pelo montador, gerando, respectivamente, os arquivos objeto `arq1.o` e `arq2.o`. Se inspecionarmos a tabela de símbolos desses dois objetos com o programa `nm`, que símbolos aparecerão na saída como **D** (símbolo na área de dados, exportado), **T** (símbolo na área de código, exportado) e **U** (referência externa) para cada um deles??

(b) Suponha que tentemos gerar um executável executando o comando ***gcc -o prog arq1.o arq2.o***. O executável seria gerado? Justifique sua resposta.

4. (1,0 ponto) Escreva o código *assembly* equivalente ao código de máquina que a função **geracod** do seu segundo trabalho geraria para a função SB a seguir.

```
v1 := p1
v2 := p2
v2 += v1
ret v2
```

Boa Prova!