Nome: Lucas Lemos
Matrícula: 2110013

1.a) O tamanho de um inteiro é de 4 bytes, e esses bytes estão ordenados em *little-endian,* ou seja, do byte menos significativo para o mais significativo.

```
(base) puc@notepuc07:~/Documentos/Pessoal/inf1018/inf1018-software-basico/LAB2$ ./lab2-ex1
0x7fffeaba3994 - 10
0x7fffeaba3995 - 27
0x7fffeaba3996 - 00
0x7fffeaba3997 - 00
```

b)c) O *long* e o *short* também são armazenados em *little-endian*, porém, o *long* ocupa 8 bytes, enquanto o *short* ocupa 2 bytes.

d) No caso de atribuir *char i = 10000*, o dump mostra apenas o primeiro byte da memória, já que um *char* ocupa apenas 1 byte.

```
(base) puc@notepuc07:~/Documentos/Pessoal/inf1018/inf1018-software-basico/LAB2$ ./lab2-ex1
0x7ffe46f6ada7 - 10
```

2.a) A função *string2num* pega um valor numérico em *string* e converte para *int*. O valor de *(*s - '0')* a cada iteração representa a diferença de cada caractere para o caractere '0'. Essa diferença vai resultar no próprio caractere. O valor de *a* representa o número que está sendo criado, à medida que avança no loop.

b)

```c
#include <ctype.h>
#include <stdio.h>

int string2num (char *s, int base) {
  int a = 0;
  for (; *s; s++){
    a = a*base + (*s - '0');
  }
  return a;
}

int main (void) {
  printf("==> %d\n", string2num("1234", 2));
  printf("==> %d\n", string2num("1234", 2) + 1);
  printf("==> %d\n", string2num("1234", 2) + string2num("1", 2));
  return 0;
}
```

c) A maior base que podemos utilizar para esse modelo de código é 62, já que podemos contar com os 10 algarismos numéricos, 26 letras minúsculas e 26 letras maiúsculas, já que possuem diferentes valores no código ASCII.

```c
#include <ctype.h>
#include <stdio.h>

int string2num (char *s, int base) {
  int a = 0;
  for (; *s; s++){
    if (isdigit(*s)){
        a = a*base + (*s - '0');
    }
    else{
        a = a*base + (*s - 'a' + 10);
    }
  }
  return a;
}

int main (void) {
  printf("==> %d\n", string2num("1234", 2));
  printf("==> %d\n", string2num("1234", 2) + 1);
  printf("==> %d\n", string2num("1234", 2) + string2num("1", 2));
  return 0;
}
```