Homework 4

Problem1

Please show the relocation entry and process of following instruction:

```
Disassembly of section .text:
0000000000000000 <main>:
    ...
    2: c7 05 00 00 00 00 ed ad 0f 00   movl$0xfaded, buf(%rip)
```

Relocation entry:

offset: 4 type: R_X86_64_PC32 symbol: buf addend: -8

Relocation:

ADDR(.text) = 0x4004d6

refaddr = ADDR(.text) + r.offset = 0x4004da

ADDR(r.symbol) = 0x601030

*refptr = ADDR(r.symbol) – refaddr + r.addend

= 0x601030 – 0x4004d6 – 0x8

= 0x200b4e

Problem 2

Given the PLT table:

```
PLT[1] <free>
400450: jmpq *0x20055a(%rip)
400456: pushq $0x0
40045b: jmpq 0x400440
PLT[2] <printf>
400460: jmpq *0x200552(%rip)
400466: pushq $0x1
40046b: jmpq 0x400440
...
PLT[5] <malloc>
400490: jmpq *0x20053a(%rip)
400496: pushq $0x4
40049b: jmpq 0x400440
```

1. Please fill in the GOT table before the execution of main

Address	Entry	Contents	Description
0x600998	GOT[0]	--	Addr of .dynamic
0x6009a0	GOT[1]	--	Addr of reloc entries
0x6009a8	GOT[2]	--	Addr of dynamic linker
0x6009b0	GOT[3]	0x400456	Entry of free
0x6009b8	GOT[4]	0x500566	Entry of printf
0x6009d0	GOT[7]	0x400496	Entry of malloc

2. We have the following code:

```
1: #include <stdlib.h>
2: void main(){
3:   int a = 1;
4:   printf("%d\n", a);
5:   char *c;
6:   c = (char*)malloc(4);
7:   printf("%d\n",a);
8:   free(c);
9: }
```

The addresses of functions are given:

printf	0x00007ffff7a81cf0
malloc	0x00007ffff7aacfc0
free	0x00007ffff7aad600

Which entry of the GOT table will change and what will it be after the execution of

line 4: GOT[4]: 0x00007ffff7a81cf0

line 6: GOT[7]: 0x00007ffff7aacfc0

line 7: no change

line 8: GOT[3]: 0x00007ffff7aad600

Problem3

Suppose we have **main.c** and a shared object **dog.so**. We want to invoke a function **void bowwow()** in dog.so.
Please complete the following code in main.c using the dynamic linking interfaces.

```
#include <stdio.h>
#include <stdlib.h>
#include <dlfcn.h>

int main(){
    // Your codes here.
    void *handle;
    void (*bowwow)();
    char *error;

    handle = dlopen("./dog.so", RTLD_LAZY);
    if(!handle){
       fprintf(stderr, "%s\n", dlerror());
       exit(1);
    }

    bowwow = dlsym(handle, "bowwow");
    if((error = dlerror()) != NULL){
       fprintf(stderr, "%s\n", dlerror());
       exit(1);
    }

    bowwow();
    return 0;
}
```