NT209.Q11.ANTN - LAB1 - 24521208

Bài thực hành 1

```c
#include <stdio.h>

int main() {
    unsigned long long a = 0xC56A8FA37DD25LL;
    unsigned long long b = 0x146C522CB8051LL;
    unsigned long long res = a ^ b;

    printf("0x%llx ^ 0x%llx = 0x%llx \n", a, b, res);

    unsigned int c = 0x28378u;
    printf("-(~0x%x) = 0x%x", c, -(~c));

    return 0;
}
```

- Output:

```
0xc56a8fa37dd25 ^ 0x146c522cb8051 = 0xd106dd8fc5d74
-(~0x28378) = 0x28379
```

- Với phép tính đầu là 2 số lớn hơn 32 bit nên ta sử dụng unsigned long long (64 bit) thay vì int (32 bit) để không bị mất bit và không có vấn đề về dấu.
- Phép tính thứ 2 có `-~c = c + 1` -> `-~0x28378 = 0x28378 + 1 = 0x28379`.

Bài thực hành 2

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

int main() {
    char s[] = {0x31, 0x32, 0x33, 0x34};
    int a = 12312312;

    char *p = (char *)(uintptr_t)a;
    printf("pointer to a: %p \n", (void *)p);

    char *ps = s;
    ps[4] = '\0';
    printf("s = %s \n", ps);
```

```
    unsigned char *p_bytes = (unsigned char *)&ps;
    for (int i = 0; i < 4; i++)
        s[i] = p_bytes[i];

    printf("pointer to s: %p \n", (void *)ps);
    printf("s after writing 4 first bytes of pointer s to s:\n");
    for (int i = 0; i < 4; i++)
        printf("0x%x ", (unsigned char)s[i]);


    return 0;
}
```

- Output:

```
pointer to a: 0xbbdef8
s = 1234
pointer to s: 0x7ffe14d28360
s after writing 4 first bytes of pointer s to s:
0x60 0x83 0xd2 0x14
```

- Pointer char *p để lưu trữ giá trị của (char *)a. Sử dụng `uintptr_t` để cast int (32 bit) sang pointer (64 bit).

```
pointer.c: In function ‘main’:
pointer.c:8:15: warning: cast to pointer from integer of different size [-
Wint-to-pointer-cast]
    8 |     char *p = (char *)a;
      |
```

- Thêm null terminator vào s để tạo thành chuỗi hoàn chỉnh.
- Sử dụng `unsigned char *p_bytes` để lấy 4 byte đầu của con trỏ s.

Bài thực hành 3

```
#include <stdio.h>
#include <stdint.h>

int main() {
    int32_t base = -2313;

    // ~-2313 = 100100001000
    // L =   000000001001100
    // o =   000000001101111
    // s =   000000001110011
    // e =   000000001100101
```

```
    // r =   000000001110010
    // SPC = 000000000100000
    // u =   000000001110101
    // s =   000000001110011
    // e =   000000001100101
    // SPC = 000000000100000
    // L =   000000001001100
    // L =   000000001001100
    // M =   000000001001101

    putchar((~base >> 5) | 4);  // L
    putchar((~base >> 5) | 39); // o
    putchar((~base >> 7) | 97); // s
    putchar((~base >> 6) | 65); // e
    putchar((~base >> 7) | 96); // r
    putchar((~base >> 7) | 97); // s
    putchar((~base >> 6) & 32); // SPC
    putchar((~base >> 6) | 81); // u
    putchar((~base >> 7) | 97); // s
    putchar((~base >> 6) | 65); // e
    putchar((~base >> 6) & 32); // SPC
    putchar((~base >> 5) | 4);  // L
    putchar((~base >> 5) | 4);  // L
    putchar((~base >> 5) | 5);  // M

    return 0;
}
```

- Với `~-2313 = 100100001000` và kí tự như `L = 000000001001100` ta shift right 5 `~-2313` để được `1001000`, ta có `0b1001000 = 72` và `0b1001100 = 76` -> `76 ^ 72 = 4`. Vậy ta được `(~-2313 >> 5) | 4 = 76` ta được kí tự `L`, làm tương tự với các kí tự khác.

Thử thách

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <time.h>

static int32_t rnd_range(int32_t lo, int32_t hi) {
    uint32_t r = ((uint32_t)rand() << 31) ^ (uint32_t)rand();
    uint32_t span = (uint32_t)(hi - lo) + 1ULL;
    r %= span;
    return (int32_t)lo + (int32_t)r;
}

static int32_t _k(int argc, char **argv) {
```

```
    if (argc < 2) {
        fprintf(stderr, "Usage: %s <number>\n", argv[0]);
        exit(2);
    }
    char *end = NULL;
    int32_t u = strtoll(argv[1], &end, 10);
    if (*end != '\0') {
        fprintf(stderr, "Invalid input! \nUsage: %s <number>\n", argv[0]);
        exit(3);
    }
    return u;
}

int main(int argc, char **argv) {
    int32_t U = _k(argc, argv);
    srand((unsigned)time(NULL));
    int32_t R = rnd_range(-100000000L, 100000000L);
    if (U < -100000000L || U > 100000000L) exit(1);

    int32_t r1 = ((R ^ U) >> 22) & -1L;
    int32_t r2 = (((~R) ^ U) >> 22) & -1L;
    int32_t A = U | 260925L;
    if (A != ~(-9173822L)) exit(-1);
    int32_t B = A << 14;
    int32_t C = (r1 | r2) & B;

    if ((uint32_t)C == 0xFECF4000u) {
        fputs("You found your beloved pinanek\n", stdout);
    }
    return 0;
}
```

- Ta thấy `r1 = (R ^ U)`, `r2 = (~R ^ U)` -> `C = (r1 | r2) = 1...1` -> `C = 1...1 & B` -> `C = B`.
- `A = U | 260925` mà `B = A << 14` -> `C = (U | 260925) << 14` .
- `C == 0xfecf4000` -> `C = (U | 260925) << 14 == 0xfecf4000`.
- `if A != ~(-9173822) exit(-1);` -> `U | 260925 != ~(-9173822)` -> `U != ~(-9173822) ^ 260925` -> `U != 8912896`.
- Vậy ta chỉ cần nhập 8912896 ta sẽ thỏa mãn được điều kiện đề bài.

```
❯ ./random_me 8912896
You found your beloved pinanek
```