báo cáo bài tập tuần 1

môi trường thực hiện: windows, linux 32 bit

Linux: sử dụng nasm để build

- nasm -f elf file.asm: biên dịch tạo file object
- ld -m elf_i386 file.o -o file: static link file object để tạo file thực thi

windows: sử dụng masm32 để build

- C:\masm32\bin\ml /c /coff file.asm: /c để ml không link với file thực thi, /coff để tạo file coff
- C:\masm32\bin\link /subsystem:console: tạo chương trình chạy trên console

Bài 1: Hello world

- **Linux:** sử dụng syscall để in chuỗi "hello world" từ section .data ra màn hình
  - section .data để lưu dữ liệu
    - tạo 1 biến **s** với mỗi ký tự là 1 byte và ký tự cuối cùng là ký tự xuống dòng
  - section .text để viết code thực thi
    - tạo 1 global lable _start làm entry point của chương trình section .text để viết code thực thi trong đó :
      - gán system_call của sys_write là 4 trong linux system call table vào **eax**, link tham khảo: Linux System Call Table (nps.edu)
      - gán file descriptor của stdout là 1 vào **ebx**
      - gán địa chỉ của **s** vào **ecx**
      - gán độ dài của **s** vào **edx**
      - ngắt chương trình bằng int 0x80
      - gán system_call của sys_exit là 1 trong linux system call table vào eax
- **Windows:** sử dụng api của masm32 để in chuỗi "hello world" từ section .data ra màn hình
  - Section .data: tương tự linux
  - Section .code: để viết code thực thi
    - Tạo main proc để làm entry point của chương trình
      - Stdout nhận offset của chuỗi *hello world* làm tham số
      - ExitProcess nhận 0 làm tham số

Bài 2: Echo

- **Linux:** sử dụng syscall để nhập từ bàn phím chuỗi từ section .bss và in ra màn hình
  - section .data để lưu dữ liệu

        - tạo 1 biến **ent** với mỗi ký tự là 1 byte và ký tự cuối cùng là ký tự xuống dòng chưa chuỗi *Enter text to echo*
    - section .bss để khai báo dữ liệu
        - tạo 1 biến **s** khai báo 32 byte dữ liệu
    - section .text để viết code thực thi
        - tạo 1 global lable _start làm entry point của chương trình section .text để viết code thực thi trong đó :
            - đầu tiên gọi lable **printEnter** để in ra màn hình chuỗi *Enter text to echo* trong đó:
                - gán system_call của sys_write là 4 trong linux system call table vào **eax**, link tham khảo: Linux System Call Table (nps.edu)
                - gán file descriptor của stdout là 1 vào **ebx**
                - gán địa chỉ của **s** vào **ecx**
                - gán độ dài của **s** vào **edx**
                - ngắt chương trình bằng int 0x80
            - tiếp theo gọi lable **inp** để nhập chuỗi **s** trong đó
                - gán system_call của sys_write là 3 trong linux system call table vào **eax**, link tham khảo: Linux System Call Table (nps.edu)
                - gán file descriptor của stdin là 0 vào **ebx**
                - gán địa chỉ của **s** vào **ecx**
                - gán độ dài của **s** vào **edx**
                - ngắt chương trình bằng int 0x80
            - cuối cùng gọi lable **outp** để in ra chuỗi **s**, cách thực hiện tương tự in ra chuỗi Enter text to echo

- **Windows:** sử dụng api của masm32 để nhập biến **s** từ section .data? ra màn hình
    - Section .data: tương tự linux
    - Section .data?: tương tự section .bss trên linux
    - Section .code: để viết code thực thi
        - Tạo main proc để làm entry point của chương trình
            - đầu tiên in ra màn hình chuỗi *Enter text to echo* trong đó:
                - StdOut nhận offset của chuỗi *hello world* làm tham số
            - Tiếp theo nhập vào biến **s** với:
                - StdIn nhận 2 tham số là số lương ký tự nhập và offset của biến **s** làm tham số
            - Cuối cùng sửu dụng StdOut để in ra màn hình biến **s**

Bài 3: Uppercase

- **Linux:** sử dụng syscall để nhập từ bàn phím chuỗi **s** từ section .bss, thực hiện chuyển sang chữ in hoa và in ra màn hình
  - section .bss để khai báo dữ liệu
    - tạo 1 biến **s** khai báo 32 byte dữ liệu
  - section .text để viết code thực thi
    - tạo 1 global lable _start làm entry point của chương trình section .text để viết code thực thi trong đó :
      - đầu tiên gọi lable **inp** để nhập chuỗi **s** trong đó
        - gán system_call của sys_write là 3 trong linux system call table vào **eax**, link tham khảo: Linux System Call Table (nps.edu)
        - gán file descriptor của stdin là 0 vào **ebx**
        - gán địa chỉ của **s** vào **ecx**
        - gán độ dài của **s** vào **edx**
        - ngắt chương trình bằng int 0x80
      - tiếp theo gọi lable uppercase để thực hiện chuyển chữ in thường sang in hoa bằng cách trích xuất từng ký tự từ **s** ra, kiểm tra xem có phải ký tự in thường không, nếu không thì công với 32 hay 0x48 cho đến ký tự 0xA thì kết thúc, ngược lại thì giữ nguyên ký tự nhập và xét ký tự tiếp theo.
      - cuối cùng gọi lable **outp** để in ra chuỗi **s**, cách thực hiện tương tự in ra chuỗi Enter text to echo
        - gán system_call của sys_write là 4 trong linux system call table vào **eax**, link tham khảo: Linux System Call Table (nps.edu)
        - gán file descriptor của stdout là 1 vào **ebx**
        - gán địa chỉ của **s** vào **ecx**
        - gán độ dài của **s** vào **edx**
        - ngắt chương trình bằng int 0x80

- **Windows:** sử dụng api của masm32 để nhập biến **s** từ section .data? ra màn hình
  - Section .data?: tương tự section .bss trên linux
  - Section .code: để viết code thực thi
    - Tạo main proc để làm entry point của chương trình
      - Đầu tiên nhập vào biến **s** với:
        - StdIn nhận 2 tham số là số lương ký tự nhập và offset của biến **s** làm tham số
      - Tiếp theo thực hiện hàm uppercase tương tự linux

- Cuối cùng sửu dụng StdOut để in ra màn hình biến **s**
    - StdOut nhận offset của biến **s** làm tham số

Bài 4: SimpleAddition

- **Linux:**
    - Section .data khai báo 4 trường dữ liệu
        - Pa chứa chuỗi "input first number "
        - Pb chứa chuỗi "input second number"
        - S chứa chuỗi "sum is"
        - Iv chứa chuỗi "invalid number"
    - section .bss để khai báo dữ liệu
        - tạo 1 biến **a** khai báo 15 byte dữ liệu
        - tạo 1 biến **b** khai báo 15 byte dữ liệu
    - section .text để viết code thực thi
        - tạo 1 global lable _start làm entry point của chương trình section .text để viết code thực thi trong đó :
            - đầu tiên gọi lable **inpa** để nhập biến **a** trong đó
                - gán system_call của sys_write là 3 trong linux system call table vào **eax**, link tham khảo: Linux System Call Table (nps.edu)
                - gán file descriptor của stdin là 0 vào **ebx**
                - gán địa chỉ của **s** vào **ecx**
                - gán độ dài của **s** vào **edx**
                - ngắt chương trình bằng int 0x80
            - tiếp theo gọi lable atoi để chuyển chuỗi a nhập vào thành số, nếu không phải số thì in ra "invalid" và thoát chương trình:
                - kiểm tra từng ký tự trong khoảng 0x30-0x39, nếu không hợp lệ thì in ra "invalid" và thoát chương trình
                - nếu hợp lệ thì trừ đi 0x30 để chuyển từ ký tự thành số tương ứng và nhân 10 rồi cộng với số tiếp để chuyển 1 chuỗi thành số tương ứng(VD: 1234 = (((1*10) +2) *10 +3)*10 + 4)
                - ngoài ra nếu số sau khi chuyển tràn số thì in ra "invalid" và thoát chương trình, kiểm tra bằng cách check cờ OF
            - làm tương tự với lable **inpb** để nhập chuỗi **b**
            - sau khi đã có 2 số từ 2 chuỗi nhập vào a và b thì thực hiện so sánh với 0x7fffffff để kiểm tra có vượt quá 31 bit không, nếu không thì cho phép cộng
            - tiếp theo thực hiện chuyển tổng 2 số đã cộng lại thành chuỗi để in ra màn hình. Thực hiện bằng cách lấy ra từng số với instruction **div** rồi cộng với 0x30 cho đến hết số.

- cuối cùng gọi lable **outp** để in ra tổng cách thực hiện tương tự in ra chuỗi Enter text to echo
    - gán system_call của sys_write là 4 trong linux system call table vào **eax**, link tham khảo: Linux System Call Table (nps.edu)
    - gán file descriptor của stdout là 1 vào **ebx**
    - gán địa chỉ của tổng 2 số vào **ecx**
    - gán độ dài của tổng 2 số vào **edx**
    - ngắt chương trình bằng int 0x80

- **Windows:**
    - Section .data: tương tự section .data trên windows
    - Section .data?: tương tự section .bss trên linux
    - Section .code: để viết code thực thi
        - Tạo main proc để làm entry point của chương trình
            - Đầu tiên gọi lable **inpa** để in ra chuỗi a với:
                - StdIn nhận 2 tham số là số lương ký tự nhập và offset của biến **a** làm tham số
            - Tiếp theo gọi hàm atoi để chuyển chữ sang số tương tự linux
            - Tiếp theo thực hiện hàm uppercase tương tự linux
            - Thực hiện cộng và chuyển chuỗi tương tự linux
            - Cuối cùng sửu dụng StdOut để in ra màn hình tổng 2 số
                - StdOut nhận offset của tổng 2 số làm tham số