School of Mathematical and Computational Sciences

Indian Association for the Cultivation of Science

Compiler Construction: COM 5202

Tutorial II (22 January, 2025)

M. Sc Semester IV: 2024-2025 *Instructor:* Goutam Biswas

Following two functions use `read()` *and* `write()` *system calls to read and write one character from the* `stdin` *and* `stdout`*.*

```
#include <unistd.h>
char readChar(){
     char c;

     read(STDIN_FILENO, &c, 1);
     return c;
}
```

```
#include <unistd.h>
void writeChar(char c){
     write(STDOUT_FILENO, &c, 1);
}
```

These calls to `read()` *and* `write()` *can be replaced by embedded inline assembly language code of x86-84 architecture with a software interrupt.*

```
char readChar(){
     char c=0;

      __asm__ __volatile__ (
     "movq $0, %%rax # 0 for read \n\t"
     "movq $0, %%rdi # 0 for stdin \n\t"
     "movq $1, %%rdx # 1 byte \n\t"
     "syscall \n\t"
     :
     :"S" (&c)
     ) ;

     return c;
}

void writeChar(char c){
     __asm__ __volatile__ (
          "movq $1, %%rax # 1 for write \n\t"
          "movq $1, %%rdi # 1 for stdout \n\t"
          "movq $1, %%rdx # 1 byte \n\t"
          "syscall \n\t"
          :
          :"S" (&c)
      ) ;
}
```

Note the following ABI specification:

(a) `rax` ← *system call code.*

(b) `rdi` ← *1st parameter.*

(c) `rsi` ← *2nd parameter.*

(d) `rdx` ← *3rd parameter.*

(e) `eax` → *return value.*

Link:
`https://chromium.googlesource.com/chromiumos/docs/+/master/constants/syscalls.md`

Also note the following inline assembly language codes for different CPU registers.

(i) `rdi`: *'D'*

(ii) `rsi`: *'S'*

(iii) `rdx`: *'d'*

(iv) `rax`: *'a'*

Link: `https://gcc.gnu.org/onlinedocs/gcc/Extended-Asm.html`

Exercise 1. **[10]**

Write the following two C functions without using any library function or `read()`, `write()` *call.*

(a) `int readFloat(float *xP)`: *reads a floating-point number of the form* $123.45, +123.45, -123.45. - .45, +.45, +40., -40., 123$ *etc.*
`(+|-)?(([0-9]+(\.)?[0-9]*)|([0-9]*(\.)?[0-9]+))`
from the `stdin` *in float *xP.*
It returns 0 (zero) for successful read and returns 1 (one) in case of an error. Assume that the total length of input does not exceed 15 characters.

(b) `int printFloat(float x)`: *prints a floating-point number on tt stdout. Returns 0 (zero) for successful print and returns 1 in case of an error. Assume that the total length does not exceed 15 characters.*

A few **input-output** *using the following* `main()` *are shown.*

```
int main(){
    float x;

    readFloat(&x);

    x = 2*x+5.0;
    printFloat(x);
    putchar('\n');
    return 0;
}
```

Input	Output	Input	Output
123	*251.0*	*-123*	*-241.0*
123.75	*252.5*	*+123.75*	*252.5*
.75	*6.5*	*-.75*	*3.5*
12.	*29.5*	*-12.*	*-19.0*
-123.456	*-241.912002563*	*123.456*	*251.9120025634*

Send the two functions in a file `<roll-no>.2.c` *to* `goutamamartya@gmail.com`.
Do not include `main()` *in the file.*

Exercise 2. *Write regular definition for the following languages.*

(a) *All strings of lowercase letters that if contains vowels, they must come in order* $a < e < i < o < u$.

(b) *C language comments consisting of string surrounded by* `/*` *and* `*/` *without an intervening* `*/`.

Exercise 3.

(a) *Design an NFA equivalent to the regular expression:* $(a|b)^*aba(a|b)^*$.

(b) *Design a DFA with 4 states equivalent to the regular expression:* $(a|b)^*aba(a|b)^*$.

(c) *Construct the DFA equivalent to the NFA of* **Ex3(a)** *by subset construction. Identify the equivalent states.*

(d) *Construct the DFA using the collection of* $'\bullet'$*-items.*