システムプログラミング1
期末レポート

氏名: 山田　敬汰 (Yamada，Keita)
学生番号: 09430559

出題日: 2019 年?月??日
提出日: 2019 年?月??日
締切日: 2019 年?月??日

1　概要

本演習では，C 言語で書かれたプログラムを機械語に変換する際に必要不可欠となるアセンブラについて，いくつかの演習課題を解くことを通じて学びを深めた．具体的には，入出力機能の実装やアセンブリ言語内に登場する用語の解説，そして，C 言語で記述されているプログラムの再現（関数の実装，素数を表示するプログラム及びその保存方法の改造），を行った．以下に，今回の授業内で実践した 5 つの演習課題についての詳しい内容を記述する．

2　課題 1-1

2.1　課題内容

教科書 A.8 節 「入力と出力」に示されている方法と，A.9 節 最後「システムコール」に示されている方法のそれぞれで ”Hello World” を表示せよ．両者の方式を比較し考察せよ．

2.2　作成したプログラム

```
 1          .data
 2          .align 2
 3  msg:
 4          .asciiz "Hello World"
 5
 6          .text
 7          .align 2
 8  main:
 9          la      $a0, msg
10          li      $v0, 4
11          syscall
12          j       $ra
13
```

2.3 考察

両者を比較した時に，システムコールを利用しないプログラムでは，入出力処理を行う際にアドレスを固定値で指定しているのに対し，システムコールを利用するプログラムでは，入出力処理はシステムコールにより OS 側に一任されているのでプログラム内で固定値のアドレスが指定されていない，という違いがある．この違いによって，システムコールを利用する方のプログラムは入出力装置についての情報を持っていなくても動作するので，入出力機器が変更されてもプログラムを（疎結合になっている）という利点がある．その他にも，OS 側が許可した動作しかできないようになっているので（各種機器とプロセッサ間を仲介し，互いが直接アクセスしないようにしている），アドレスを直接指定するよりもセキュアである．

3 課題 1-2

3.1 課題内容

アセンブリ言語中で使用する .data， .text および .align とは何か解説せよ．下記コード中の 6 行目の .data がない場合，どうなるかについて考察せよ．

```
1:          .text
2:          .align  2
3: _print_message:
4:          la      $a0, msg
5:          li      $v0, 4
6:          .data
7:          .align  2
8: msg:
9:          .asciiz "Hello!!\n"
10:          .text
11:          syscall
12:          j       $ra
13: main:
14:          subu    $sp, $sp, 24
15:          sw      $ra, 16($sp)
16:          jal     _print_message
17:          lw      $ra, 16($sp)
18:          addu    $sp, $sp, 24
19:          j       $ra
```

3.2 解答

.data，.text および .align とは「アセンブラ指令」と呼ばれるもので，主にプログラムの実行前に行われる前処理を制御するものである．つまり，これらの記述は機械語としてメモリに変換される訳ではない．その中でも，.data と .text はメモリ中のどこに機械語を配置するかを制御している．具体的には .data はデータをデータセグメントに配置することを指示し，.text はデータをテキストセグメントに配置することを指示している．何故，データセグメントとテキストセグ

メントを区別する必要があるのかというと，データセグメントの内容には読み込みと書き込みが両方行われるのに対し，テキストセグメントの内容は読み込みしか行われない（値が不変）ので，同一プログラムを他のプロセスで実行する時にテキスト部分を共有することができたり，読み込み専用領域にデータを置くことができる，というメリットがあるからである．

また .align はオプションとして整数 n を指定し，データが 2^n ずつの領域に確保される様に余白を取るための命令である．何故余白を取る必要があるのかというと，MIPS は 32 ビット（4 バイト）ずつデータを CPU とやり取りするため，余白を取って（この場合は $n = 2$）データを綺麗に整列させておくことで，アクセス回数を減らすことが可能となり，システムの効率化に繋がるからである．

最後に，上記プログラム内から .data がない場合にどうなるかというと，

4 課題 1-3

4.1 課題内容

教科書 A.6 節 「手続き呼出し規約」に従って，関数 fact を実装せよ．(以降の課題においては，この規約に全て従うこと) fact を C 言語で記述した場合は，以下のようになるであろう．

```
 1: main()
 2: {
 3:   print_string("The factorial of 10 is ");
 4:   print_int(fact(10));
 5:   print_string("\n");
 6: }
 7:
 8: int fact(int n)
 9: {
10:   if (n < 1)
11:     return 1;
12:   else
13:     return n * fact(n - 1);
14: }
```

5 課題 1-4

5.1 課題内容

素数を最初から 100 番目まで求めて表示する MIPS のアセンブリ言語プログラムを作成してテストせよ．その際，素数を求めるために下記の 2 つのルーチンを作成すること．

- test_prime(n) n が素数なら 1，そうでなければ 0 を返す
- main() 整数を順々に素数判定し，100 個プリント

5.1.1 C 言語で記述したプログラム例

```
 1: int test_prime(int n)
 2: {
 3:   int i;
 4:   for (i = 2; i < n; i++){
 5:     if (n % i == 0)
 6:       return 0;
 7:   }
 8:   return 1;
 9: }
10:
11: int main()
12: {
13:   int match = 0, n = 2;
14:   while (match < 100){
15:     if (test_prime(n) == 1){
16:       print_int(n);
17:       print_string(" ");
18:       match++;
19:     }
20:     n++;
21:   }
22:   print_string("\n");
23: }
```

6 課題 1-5

6.1 課題内容

素数を最初から 100 番目まで求めて表示する MIPS のアセンブリ言語プログラムを作成してテストせよ．ただし，配列に実行結果を保存するように main 部分を改造し，ユーザの入力によって任意の番目の配列要素を表示可能にせよ．

6.1.1 C 言語で記述したプログラム例

```
1: int primes[100];
2: int main()
3: {
4:   int match = 0, n = 2;
5:   while (match < 100){
6:     if (test_prime(n) == 1){
7:       primes[match++] = n;
8:     }
9:     n++;
```

```
10:   }
11:   for (;;){
12:     print_string("> ");
13:     print_int(primes[read_int() - 1]);
14:     print_string("\n");
15:   }
16: }
```

7 感想

8 作成したプログラム