システムプログラミングレポート
演習課題:

09426532 高橋　桃花

出題日: 2015 年 4 月 13 日
提出日: 2015 年 7 月 13 日
締切日: 2015 年 7 月 13 日

1　概要

1.1　課題 1-1

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．

1.2　課題 1-2

アセンブリ言語中で使用する.data, .text 及び.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．

1.3　課題 1-3

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．

1.4　課題 1-4

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．

1.5　課題 1-5

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．

2　プログラムの作成方針

3　重視したこと

4　プログラムリスト

4.1　課題 1-1

```
.text
.align 2
```

```
main:
move $s0, $ra # main を呼んだ戻り先のアドレスが入っている
# $ra を$s0 に保存しておく

li $a0, 72 # 'H' = 72, $a0 = H
jal putc # jamp and link to putc
li $a0, 101 # 'a' = 101, $a0 = e
jal putc #
li $a0, 108 # 'l' = 108, $a0 = l
jal putc #
li $a0, 108 # 'l' = 108, $a0 = l
jal putc #
li $a0, 111 # 'o' = 111, $a0 = o
jal putc #
la $a0, 32
jal putc # print " "
li $a0, 87 # 'W' = 87, $a0 = W
jal putc #
li $a0, 111 # 'o' = 111, $a0 = o
jal putc #
li $a0, 114 # 'r' = 114, $a0 = r
jal putc #
li $a0, 108 # 'l' = 108, $a0 = l
jal putc #
li $a0, 100 # 'd' = 100, $a0 = d
jal putc #

move $ra, $s0 # $s0 に保存しておいた戻り先を$ra に入れる
j $ra # main を呼んだ戻り先に飛ぶ
putc:
lw $t0, 0xffff0008 # $t0 = *(oxffff0008)
li $t1, 1 # $t1 = 1
and  $t0, $t0, $t1 # $t0 =& $t1
beqz $t0, putc # if( $t0==c ) goto putc
sw $a0, 0xffff000c # *(0xffff000c) = $a0
j $ra
```

4.2 課題 1-2

```
.data
.align 2
str:
.asciiz "Hello World" # "Hello World"の文字列を定義
```

```
.text
.align 2
main:
li $v0, 4 # print_string のシステム・コール・コード
la $a0, str  # プリントする文字列
syscall # str を出力
```

4.3 課題 1-3

5 プログラムの使用法

6 プログラムの作成過程に関する考察

7 得られた結果に関する考察，あるいは設問に対する回答

7.1 課題 1-1

.data: 文字列をプログラムのデータ・セグメントに格納する．データ・セグメントとは，ソールファイル中のデータの 2 進数表現を保持するオブジェクトファイルの 1 セクションである．

.text: 命令をテキスト・セグメントに格納する．テキスト・セグメントとは，ソースファイル中のルーチンに対応する機械語コードを保持する．

.align: