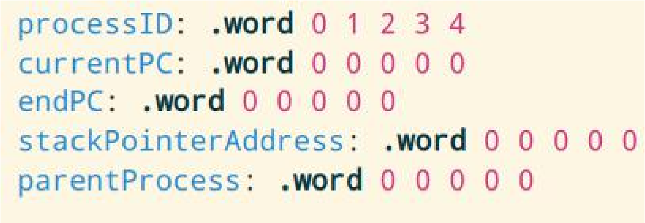
Can BEYAZNAR
161044038

CSE312 HOMEWORK 3 REPORT

1) Problem Solution

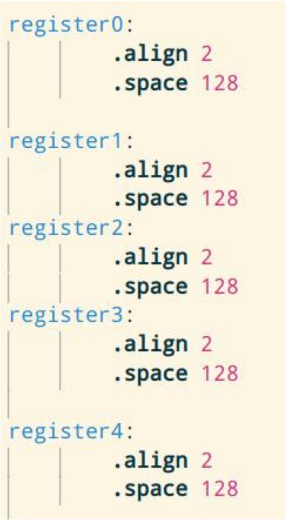
A) SPIMOS_GTU_X.S PART

In this assignment, I firstly defined the parameters that should be in my process table in asm. These parameters are as follows.



```
processID: .word 0 1 2 3 4
currentPC: .word 0 0 0 0 0
endPC: .word 0 0 0 0 0
stackPointerAddress: .word 0 0 0 0 0
parentProcess: .word 0 0 0 0 0
```

I created parameters that keep the address of this data in syscall.cpp file so that these parameters can be updated and written during interrupt. So I can update the data in the process table with the set_mem_word function. Then, to be able to handle an interrupt inside the asm, the spimtimer_handler must go to that part of the asm whenever there is an interrupt. To do this, I called a syscall, pointing to the interrupt_handler label. So when an interrupt occurs, spimtimer_handler uses the parameter from syscall to go to this label. And asm handles interrupt. After all these processes are finished, we come to the call processes section. First of all, since each process will have its own register values, I created a separate register parameter for each (as follows).



```
register0:
        .align 2
        .space 128

register1:
        .align 2
        .space 128
register2:
        .align 2
        .space 128
register3:
        .align 2
        .space 128

register4:
        .align 2
        .space 128
```

For example, since SPIMOS_GTU_1.s will run a total of 4 (+1 init processes) processes, I kept 5 register parameters. Thus, the programs can continue where they left off during the backup and update stages. In the process loading section, I first upload the address of the register parameter to be used in the $ a0 register. then I load the currentPC and endPC parameters into the $ 15 and $ 24 registers. These will keep the start and end addresses of the asm file that will run. And with the help of execve, I load the asm file. I repeat these processes for all processes. And the program is waiting in an endless loop until it is interrupt.

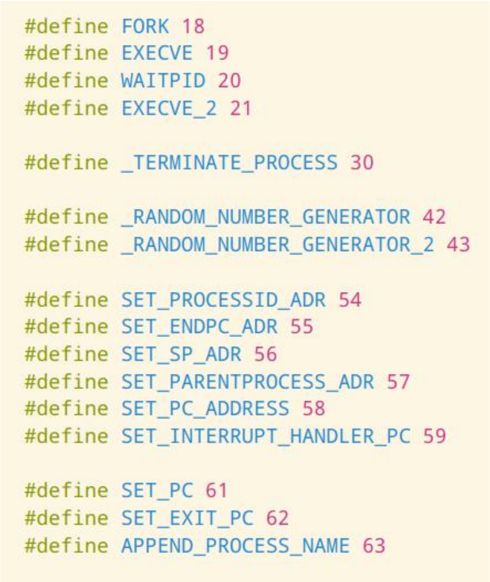
Then, in the interrupt_handler label, the register information of the process that was previously running in the $ s6 register is displayed. In the $ s7 register, the register information that will now be kept is kept. In this label, the registration information of the previously running process is recorded and the information of the new process to be loaded is loaded one by one.

NOTE: These registers are specially selected. If other asm files use these registers, the program may run incorrectly. In addition, WAITPID syscall is used in order not to interrupt the processes performed by kernels.

B) SPIM_timerHandler()

In this function, I use the parameter addresses sent by kernel to update the process table information. I use syscalls to get this information. I describe my Syscalls in detail in Syscall.cpp. Coming to SPIM_timerHandler function, I first check the waitpid_signal parameter. This parameter states that asm files are at a critical point while running. If the value of this parameter is true, the program will resume even if interrupt occurs. In other words, context switch will not be realized in any way. If false, the other process should work. To do this, I first check whether all processes are finished. If all processes are finished, I am free all the parameters that I reserved in memory with malloc. And I put the PC at the address of the kernel's exit label. Thus, the program exits safely. If all processes are not finished, I continue my process. As I mentioned while explaining the kernel part, I update the process table information in asm to be able to backup the running process. I do this with the help of currentProcess. currentProcess keeps track of which process is running. I put the PC and register information of the running process in the $ 26 and $ 22 registers. Then, if this process is not finished, I switch from "Running" to "Ready". Then I use the Round robin scheduling algorithm to run the other process. And by looking at the state of the processes, I select the process that will be running next by looking at which ones are ready. And I assign to $ 23 and $ 27 registers to update the register and PC of the process that will run in kernel. And I assign the InterruptHandlerPC to the PC so that the kernel can handle the interrupt. Thus, by going to the kernel interrupt_handler label, it can update the information by using the registers it receives from spim_timerhandler.

The syscalls I use are as follows:

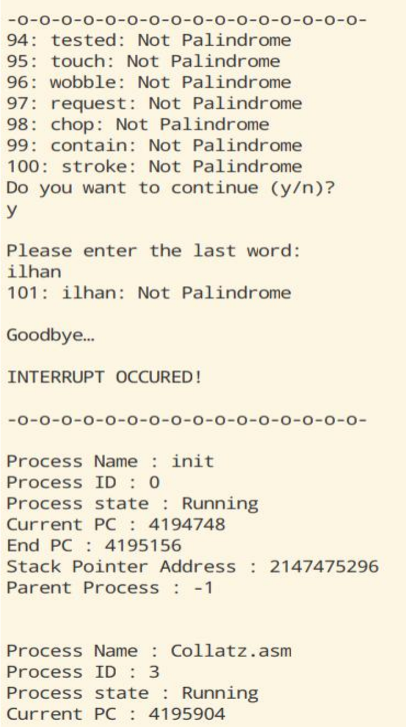


```
#define FORK 18
#define EXECVE 19
#define WAITPID 20
#define EXECVE_2 21

#define _TERMINATE_PROCESS 30

#define _RANDOM_NUMBER_GENERATOR 42
#define _RANDOM_NUMBER_GENERATOR_2 43

#define SET_PROCESSID_ADR 54
#define SET_ENDPC_ADR 55
#define SET_SP_ADR 56
#define SET_PARENTPROCESS_ADR 57
#define SET_PC_ADDRESS 58
#define SET_INTERRUPT_HANDLER_PC 59

#define SET_PC 61
#define SET_EXIT_PC 62
#define APPEND_PROCESS_NAME 63
```

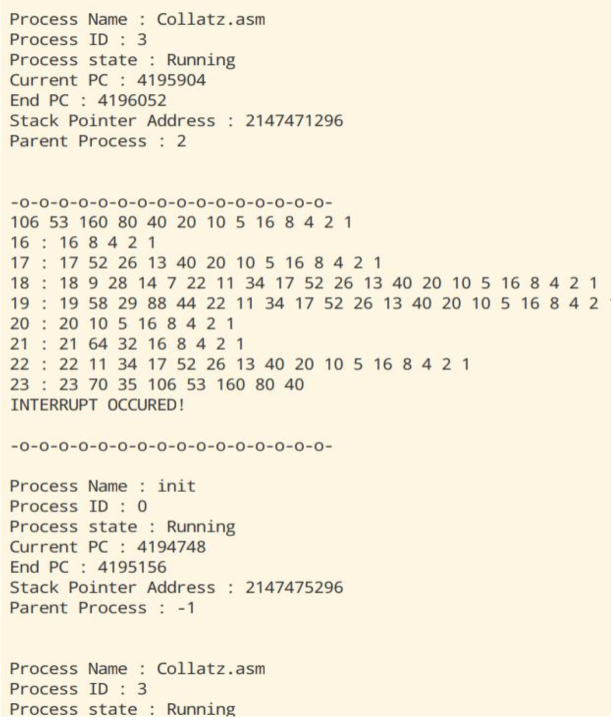
2) Results

Small examples of printouts are as follows

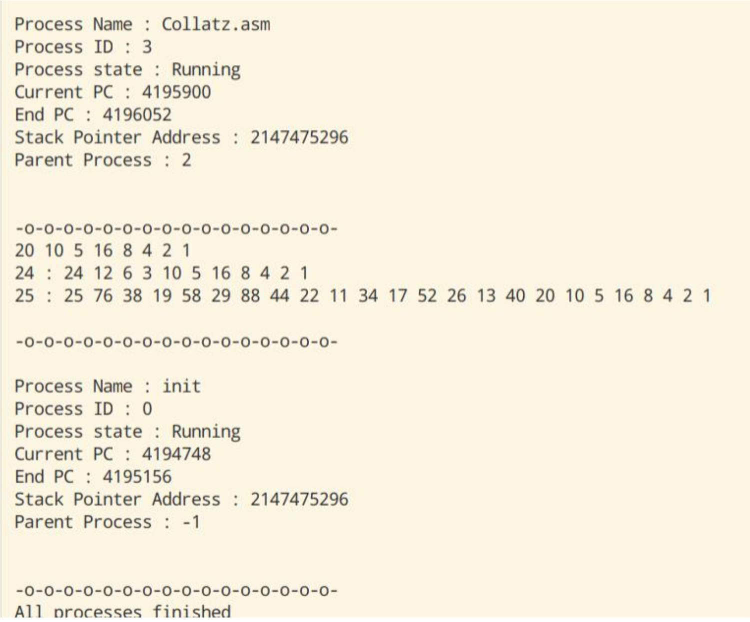
A) SPIMOS_GTU_1.s



```
-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
94: tested: Not Palindrome
95: touch: Not Palindrome
96: wobble: Not Palindrome
97: request: Not Palindrome
98: chop: Not Palindrome
99: contain: Not Palindrome
100: stroke: Not Palindrome
Do you want to continue (y/n)?
y

Please enter the last word:
ilhan
101: ilhan: Not Palindrome

Goodbye...

INTERRUPT OCCURED!

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

Process Name : init
Process ID : 0
Process state : Running
Current PC : 4194748
End PC : 4195156
Stack Pointer Address : 2147475296
Parent Process : -1


Process Name : Collatz.asm
Process ID : 3
Process state : Running
Current PC : 4195904
```

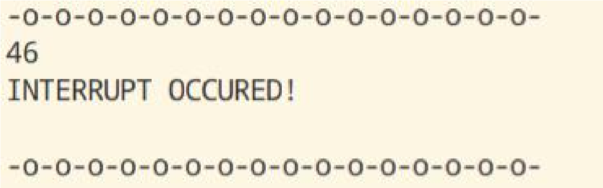


```
Process Name : Collatz.asm
Process ID : 3
Process state : Running
Current PC : 4195904
End PC : 4196052
Stack Pointer Address : 2147471296
Parent Process : 2


-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
106 53 160 80 40 20 10 5 16 8 4 2 1
16 : 16 8 4 2 1
17 : 17 52 26 13 40 20 10 5 16 8 4 2 1
18 : 18 9 28 14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
19 : 19 58 29 88 44 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2
20 : 20 10 5 16 8 4 2 1
21 : 21 64 32 16 8 4 2 1
22 : 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
23 : 23 70 35 106 53 160 80 40
INTERRUPT OCCURED!

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

Process Name : init
Process ID : 0
Process state : Running
Current PC : 4194748
End PC : 4195156
Stack Pointer Address : 2147475296
Parent Process : -1


Process Name : Collatz.asm
Process ID : 3
Process state : Running
```

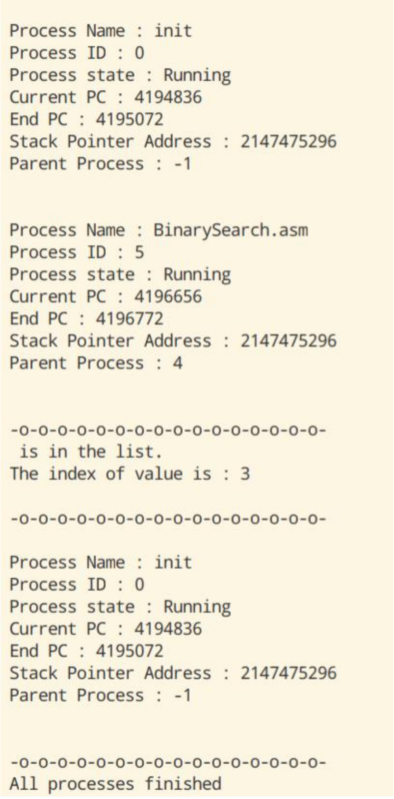


```
Process Name : Collatz.asm
Process ID : 3
Process state : Running
Current PC : 4195900
End PC : 4196052
Stack Pointer Address : 2147475296
Parent Process : 2


-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
20 10 5 16 8 4 2 1
24 : 24 12 6 3 10 5 16 8 4 2 1
25 : 25 76 38 19 58 29 88 44 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

Process Name : init
Process ID : 0
Process state : Running
Current PC : 4194748
End PC : 4195156
Stack Pointer Address : 2147475296
Parent Process : -1


-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
All processes finished
```

B) SPIMOS_GTU_2.s

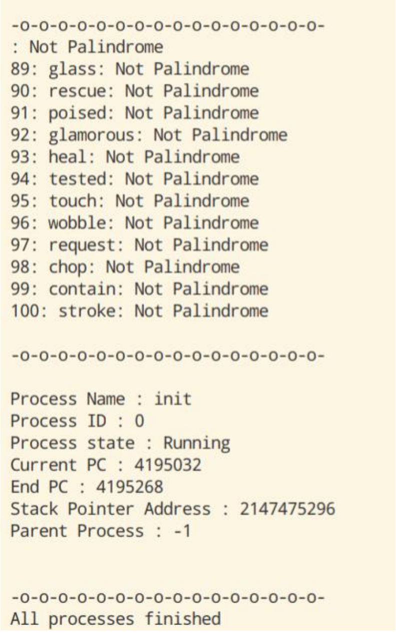


```
-O-O-O-O-O-O-O-O-O-O-O-O-O-O-O-O-O-
46
INTERRUPT OCCURED!


-O-O-O-O-O-O-O-O-O-O-O-O-O-O-O-O-O-
```



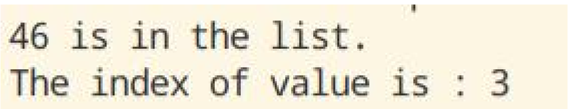
```
Process Name : init
Process ID : 0
Process state : Running
Current PC : 4194836
End PC : 4195072
Stack Pointer Address : 2147475296
Parent Process : -1


Process Name : BinarySearch.asm
Process ID : 5
Process state : Running
Current PC : 4196656
End PC : 4196772
Stack Pointer Address : 2147475296
Parent Process : 4


-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
 is in the list.
The index of value is : 3

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

Process Name : init
Process ID : 0
Process state : Running
Current PC : 4194836
End PC : 4195072
Stack Pointer Address : 2147475296
Parent Process : -1


-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
All processes finished
```

C) SPIMOS_GTU_3.s



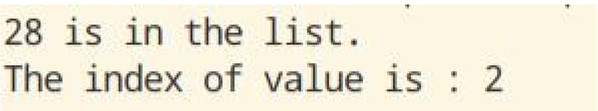
```
-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
: Not Palindrome
89: glass: Not Palindrome
90: rescue: Not Palindrome
91: poised: Not Palindrome
92: glamorous: Not Palindrome
93: heal: Not Palindrome
94: tested: Not Palindrome
95: touch: Not Palindrome
96: wobble: Not Palindrome
97: request: Not Palindrome
98: chop: Not Palindrome
99: contain: Not Palindrome
100: stroke: Not Palindrome

-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-

Process Name : init
Process ID : 0
Process state : Running
Current PC : 4195032
End PC : 4195268
Stack Pointer Address : 2147475296
Parent Process : -1


-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-o-
All processes finished
```

D) BinarySearch.asm



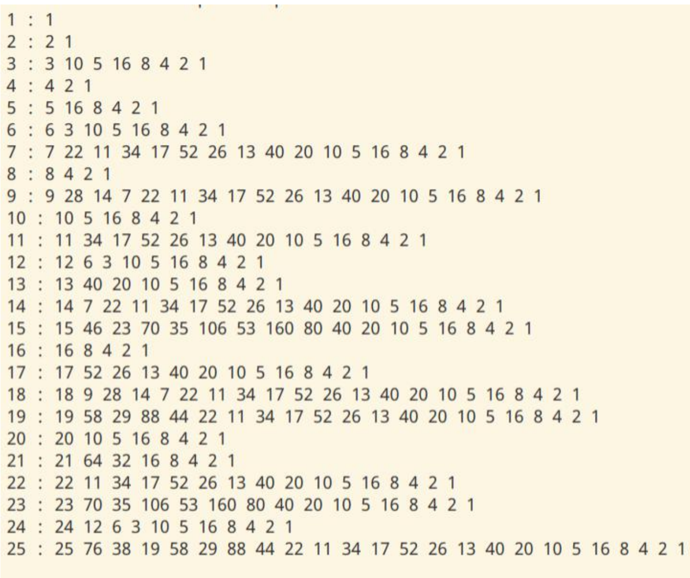
```
46 is in the list.
The index of value is : 3
```

E) LinearSearch.asm



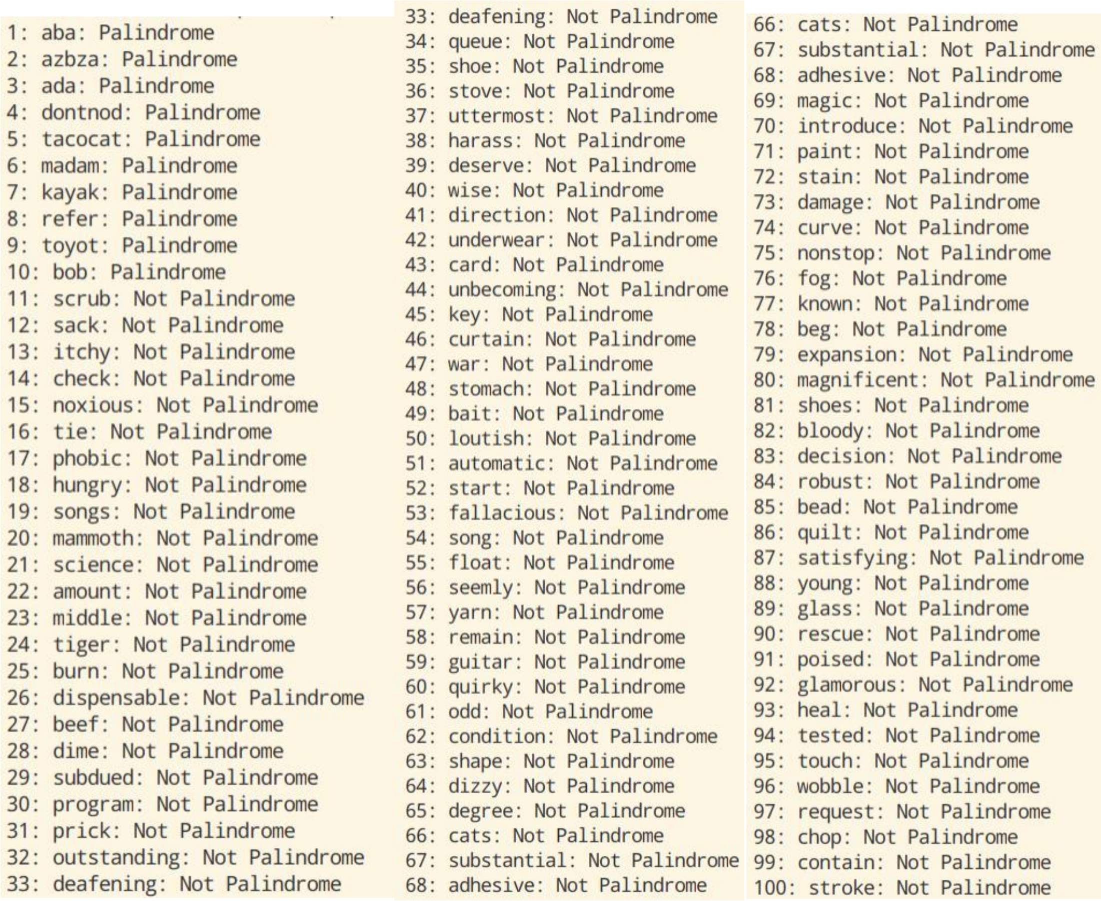
```
28 is in the list.
The index of value is : 2
```

F) Collatz.asm



```
1 : 1
2 : 2 1
3 : 3 10 5 16 8 4 2 1
4 : 4 2 1
5 : 5 16 8 4 2 1
6 : 6 3 10 5 16 8 4 2 1
7 : 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
8 : 8 4 2 1
9 : 9 28 14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
10 : 10 5 16 8 4 2 1
11 : 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
12 : 12 6 3 10 5 16 8 4 2 1
13 : 13 40 20 10 5 16 8 4 2 1
14 : 14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
15 : 15 46 23 70 35 106 53 160 80 40 20 10 5 16 8 4 2 1
16 : 16 8 4 2 1
17 : 17 52 26 13 40 20 10 5 16 8 4 2 1
18 : 18 9 28 14 7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
19 : 19 58 29 88 44 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
20 : 20 10 5 16 8 4 2 1
21 : 21 64 32 16 8 4 2 1
22 : 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
23 : 23 70 35 106 53 160 80 40 20 10 5 16 8 4 2 1
24 : 24 12 6 3 10 5 16 8 4 2 1
25 : 25 76 38 19 58 29 88 44 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1
```

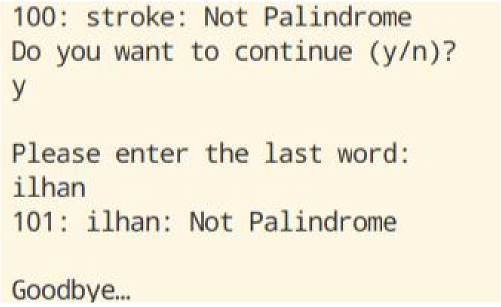
G) Palindrome.asm



```
1: aba: Palindrome
2: azbza: Palindrome
3: ada: Palindrome
4: dontnod: Palindrome
5: tacocat: Palindrome
6: madam: Palindrome
7: kayak: Palindrome
8: refer: Palindrome
9: toyot: Palindrome
10: bob: Palindrome
11: scrub: Not Palindrome
12: sack: Not Palindrome
13: itchy: Not Palindrome
14: check: Not Palindrome
15: noxious: Not Palindrome
16: tie: Not Palindrome
17: phobic: Not Palindrome
18: hungry: Not Palindrome
19: songs: Not Palindrome
20: mammoth: Not Palindrome
21: science: Not Palindrome
22: amount: Not Palindrome
23: middle: Not Palindrome
24: tiger: Not Palindrome
25: burn: Not Palindrome
26: dispensable: Not Palindrome
27: beef: Not Palindrome
28: dime: Not Palindrome
29: subdued: Not Palindrome
30: program: Not Palindrome
31: prick: Not Palindrome
32: outstanding: Not Palindrome
33: deafening: Not Palindrome
```

```
33: deafening: Not Palindrome
34: queue: Not Palindrome
35: shoe: Not Palindrome
36: stove: Not Palindrome
37: uttermost: Not Palindrome
38: harass: Not Palindrome
39: deserve: Not Palindrome
40: wise: Not Palindrome
41: direction: Not Palindrome
42: underwear: Not Palindrome
43: card: Not Palindrome
44: unbecoming: Not Palindrome
45: key: Not Palindrome
46: curtain: Not Palindrome
47: war: Not Palindrome
48: stomach: Not Palindrome
49: bait: Not Palindrome
50: loutish: Not Palindrome
51: automatic: Not Palindrome
52: start: Not Palindrome
53: fallacious: Not Palindrome
54: song: Not Palindrome
55: float: Not Palindrome
56: seemly: Not Palindrome
57: yarn: Not Palindrome
58: remain: Not Palindrome
59: guitar: Not Palindrome
60: quirky: Not Palindrome
61: odd: Not Palindrome
62: condition: Not Palindrome
63: shape: Not Palindrome
64: dizzy: Not Palindrome
65: degree: Not Palindrome
66: cats: Not Palindrome
67: substantial: Not Palindrome
68: adhesive: Not Palindrome
```

```
66: cats: Not Palindrome
67: substantial: Not Palindrome
68: adhesive: Not Palindrome
69: magic: Not Palindrome
70: introduce: Not Palindrome
71: paint: Not Palindrome
72: stain: Not Palindrome
73: damage: Not Palindrome
74: curve: Not Palindrome
75: nonstop: Not Palindrome
76: fog: Not Palindrome
77: known: Not Palindrome
78: beg: Not Palindrome
79: expansion: Not Palindrome
80: magnificent: Not Palindrome
81: shoes: Not Palindrome
82: bloody: Not Palindrome
83: decision: Not Palindrome
84: robust: Not Palindrome
85: bead: Not Palindrome
86: quilt: Not Palindrome
87: satisfying: Not Palindrome
88: young: Not Palindrome
89: glass: Not Palindrome
90: rescue: Not Palindrome
91: poised: Not Palindrome
92: glamorous: Not Palindrome
93: heal: Not Palindrome
94: tested: Not Palindrome
95: touch: Not Palindrome
96: wobble: Not Palindrome
97: request: Not Palindrome
98: chop: Not Palindrome
99: contain: Not Palindrome
100: stroke: Not Palindrome
```



```
100: stroke: Not Palindrome
Do you want to continue (y/n)?
y

Please enter the last word:
ilhan
101: ilhan: Not Palindrome

Goodbye...
```

3) ISSUES

A) In some cases, the values of Collatz write unexpectedly different. I think this is due to a break in a critical place.

B) In some cases, Palindrome may print the words it reads incompletely. Palindrome may work incorrectly because it runs too many in SPIMOS_GTU_2 and 3.

C) The user should never enter extended ascii as input. Input must be valid. Error control is not done.