Computer Organization Project #1

Menu :

```
main:
#print Menu Message
li $v0, 4
la $a0, menumsg
syscall
#Take menu input number
li $v0, 5
syscall
move $t0,$v0
#Move to the functions
beq $t0, 5, exit
beq $t0, 1, palindrome
beq $t0, 2, reversevowels
beq $t0, 3, squareFree
```

Thanks to the code on the left side, users are first informed about the menu and then they choose which program they want to run.

```
Main Menu:
1. Find Palindrome
2. Reverse Vowels
3. Find Distinct Prime
4. Lucky Number
5. Exit
Please select an option:
```

```
.data
menumsg: .asciiz "Main Menu:\n1. Find Palindrome\n2. Reverse Vowels\n3. Find Distinct Prime\n4. Lucky Number\n5. Exit\nPlease select an option: "
```

2) Reverse Vowel

```
loopReverseVowel:
  lb $t1, ($a1)     # load a byte f
  beq $t1, 0, endReverseVowelLoop
  addi $a1, $a1, 1 # increment the
  addi $t0, $t0, 1 # increment the
  #compare ASCII codes of characte
  li $t7,97 #ASCII code of a
  beq $t7,$t1,recordVowel
  li $t7,65 #ASCII code of A
  beq $t7,$t1,recordVowel
  li $t7,101 #ASCII code of e
  beq $t7,$t1,recordVowel
  li $t7,69 #ASCII code of E
  beq $t7,$t1,recordVowel
  li $t7,105 #ASCII code of i
  beq $t7,$t1,recordVowel
  li $t7,73 #ASCII code of I
  beq $t7,$t1,recordVowel
  li $t7,111 #ASCII code of o
  beq $t7,$t1,recordVowel
  li $t7,79 #ASCII code of O
  beq $t7,$t1,recordVowel
  li $t7,117 #ASCII code of u
  beq $t7,$t1,recordVowel
  li $t7,85 #ASCII code of U
  beq $t7,$t1,recordVowel
  j loopReverseVowel
```

```
reversevowels:
#Print Input: text
li $v0,4
la $a0,palmsg
syscall
#Get input from th
li $v0, 8
la $a0, userInput
li $a1, 100
syscall
#a1 = userinput ,
move $a1,$a0
```

We take input from the user and save this input in variable a1

We check each letter of the input in $a1 one by one with ASCII codes and see if it is a vowel.

If it is a vowel, we pass to the recordVowel function.

ASCII Table :

a = 97 , A = 65

e = 101 , E = 69

i = 105 , I = 73

o = 111 , O = 79

u = 117 , U = 85

```
recordVowel:
  addi $s1,$s1,1  #s1 = s1 + 1 (how many vowels)
  sb $t7, vowelArray($s5)  #store characters ascii code into array
  addi $s5, $s5, 4
  j loopReverseVowel
```

In the recordVowel function, we store the ASCII codes of the characters in "vowelArray" and we keep the number of vowels in $s1.

```
changeOrders:
beq $t3,$t0,jumpMain
addi $t3,$t3,1
lb $t1, ($a1)
addi $a1,$a1,1
  #compare ASCII codes of characters
  li $t7,97 #ASCII code of a
  beq $t7,$t1,lastvowel
  li $t7,65 #ASCII code of A
  beq $t7,$t1,lastvowel
  li $t7,101 #ASCII code of e
  beq $t7,$t1,lastvowel
  li $t7,69 #ASCII code of E
  beq $t7,$t1,lastvowel
  li $t7,105 #ASCII code of i
  beq $t7,$t1,lastvowel
  li $t7,73 #ASCII code of I
  beq $t7,$t1,lastvowel
  li $t7,111 #ASCII code of o
  beq $t7,$t1,lastvowel
  li $t7,79 #ASCII code of O
  beq $t7,$t1,lastvowel
  li $t7,117 #ASCII code of u
  beq $t7,$t1,lastvowel
  li $t7,85 #ASCII code of U
  beq $t7,$t1,lastvowel
li $v0, 11
move $a0, $t1
syscall
j changeOrders
```

We print the output to the screen in the changeOrders function.

We print the characters in the array where we saved the input, one by one.

If we come across a vowel, we print the last character in our "vowelArray" array, where we recorded all the vowels before.

If we come to the vowel for the second time, we print the second vowel from the last in the array.

if we come across a silent letter, we print it directly to the screen.

For example, our word is "Hello World".

vowelArray: {e,o,o}

We check first letter of “Hello World”. It is H, it is not a vowel so we print it to screen.

Second letter is “e” , it is vowel so we don’t print it , instead we write the last character in vowelArray.

Then we skip third and fourth letters because they are not vowel.

Fifth letter is “o” , this time instead of typing the letter "o", we print the second letter from the last in the vowelArray.

In this way, instead of the 8th letter "o", we write the 3rd letter from the end of the vowelArray "e".

3) Find Distinct Prime

```
squareFree:
#Print message
li $v0, 4
la $a0, squaremsg
syscall
#Take integer input to t0
li $v0, 5
syscall
move $t0,$v0 #t0 = Input
move $s7,$t0
```

First of all, we want the user to enter an integer value as an input, thanks to the code on the left.

```
srl $t7,$s7,1  #t7 == Input / 2
addi $t6,$zero,2  # i = 2
```

To use it in the loop, we create the t6 value and start it from 2. Since our value will start from 2 and progress to “input” , we save the “input” value in t7.

```
SquareLoop:
beq $t6, $t7, squarefreeTrue
add $t1,$zero,$t0
div $t1, $t1,$t6    #t0'i i'ye böl
mfhi $t5
beq $t5, $zero, divided
addi $t6,$t6,1
j SquareLoop
```

The working logic of our algorithm is that : by looking at whether the given number is divided 2 times by any number greater than 2 and less than input. If our input can be divided by any number, it is sent to the divided function and the divided state is checked to see if it can be divided by that number again.

```
divided:
div $t1, $t1,$t6    #t1'i i'ye böl
mfhi $t5
beq $t5, $zero, notsquare
sb $t6, myArray($s5)
addi $s3, $s3, 1
addi $s5, $s5, 4
div $t0, $t0,$t6
j SquareLoop
```

Let's take the number 27 for example. We start at 2 and try dividing. Since we cannot divide 27 by 2, we pass. There are 3 in line. Since we can divide 27 by 3, we switch to the divided function. Inside the Divided function, we are trying to divide the remaining 9 (27/3) by 3 again. Since we can divide 9 by 3 again, we understand that it is not square-free. Because we were able to divide 27 by 3 twice.

```
divided:
div $t1, $t1,$t6    #t1'i i'ye böl
mfhi $t5
beq $t5, $zero, notsquare
sb $t6, myArray($s5)
addi $s3, $s3, 1
addi $s5, $s5, 4
div $t0, $t0,$t6
j SquareLoop
```

We use "myArray" to store distinct prime factors for square-free numbers and "$s3" to calculate how many distinct prime factors there are.

```
while:
beq $s2, $s3, jumpMain
lb $s4, myArray($s5)
addi $s5, $s5, 4
li $v0, 1
add  $a0, $s4, 0
syscall
li $v0, 4
la $a0, space
syscall
addi $s2, $s2, 1
la $a0, space
j while
```

In the while loop, we print the distinct prime factor values registered in "myArray" to the screen.