CS231 lab3 Part-1

Suryansh patidar - 22B1036

Contents

1 Q1

1.1 Program 1

```
0000000000401146 <main>:
  401146:       55                      push   rbp
  401147:       48 89 e5                mov    rbp,rsp
  40114a:       48 83 ec 30             sub    rsp,0x30
  40114e:       89 7d dc                mov    DWORD PTR [rbp-0x24
    ],edi
  401151:       48 89 75 d0             mov    QWORD PTR [rbp-0x30
    ],rsi
  401155:       bf 08 20 40 00          mov    edi,0x402008
  40115a:       b8 00 00 00 00          mov    eax,0x0
  40115f:       e8 dc fe ff ff          call   401040 <printf@plt>

```

Basically this abive code is for prompt message which is "Enter three or more numbers (Terminate with CTRL + D)"

```
  401164:       c7 45 fc 00 00 00 00    mov    DWORD PTR [rbp-0x4
    ],0x0
  40116b:       c7 45 f4 00 00 00 00    mov    DWORD PTR [rbp-0xc
    ],0x0
  401172:       c6 45 f3 01             mov    BYTE PTR [rbp-0xd],0
    x1
  401176:       eb 67                   jmp    4011df <main+0x99>
  401178:       83 45 f4 01             add    DWORD PTR [rbp-0xc
    ],0x1
  40117c:       83 7d f4 01             cmp    DWORD PTR [rbp-0xc
    ],0x1
  401180:       7e 45                   jle    4011c7 <main+0x81>
  401182:       8b 45 f8                mov    eax,DWORD PTR [rbp-0
    x8]
  401185:       89 45 ec                mov    DWORD PTR [rbp-0x14
    ],eax
  401188:	8b 45 fc             	mov    eax,DWORD PTR [rbp-0x4]
  40118b:	48 98                	cdqe
  40118d:	8b 4c 85 e4          	mov    ecx,DWORD PTR [rbp+rax*4-0x1c]
  401191:	8b 45 fc             	mov    eax,DWORD PTR [rbp-0x4]
  401194:	8d 50 01             	lea    edx,[rax+0x1]
  401197:	89 d0                	mov    eax,edx
  401199:	c1 f8 1f             	sar    eax,0x1f
  40119c:	c1 e8 1f             	shr    eax,0x1f
  40119f:	01 c2                	add    edx,eax
  4011a1:	83 e2 01             	and    edx,0x1
  4011a4:	29 c2                	sub    edx,eax
  4011a6:	89 d0                	mov    eax,edx
  4011a8:	48 98                	cdqe
  4011aa:	8b 44 85 e4          	mov    eax,DWORD PTR [rbp+rax*4-0x1c]
  4011ae:	29 c1                	sub    ecx,eax
  4011b0:	89 ca                	mov    edx,ecx
  4011b2:	89 55 f8             	mov    DWORD PTR [rbp-0x8],edx
  4011b5:	83 7d f4 02          	cmp    DWORD PTR [rbp-0xc],0x2
  4011b9:	7e 0c                	jle    4011c7 <main+0x81>
  4011bb:	8b 45 ec             	mov    eax,DWORD PTR [rbp-0x14]
  4011be:	3b 45 f8             	cmp    eax,DWORD PTR [rbp-0x8]
  4011c1:	74 04                	je     4011c7 <main+0x81>
  4011c3:	c6 45 f3 00          	mov    BYTE PTR [rbp-0xd],0x0
  4011c7:	8b 45 fc             	mov    eax,DWORD PTR [rbp-0x4]
  4011ca:	8d 50 01             	lea    edx,[rax+0x1]
  4011cd:	89 d0                	mov    eax,edx
  4011cf:	c1 f8 1f             	sar    eax,0x1f
  4011d2:	c1 e8 1f             	shr    eax,0x1f
  4011d5:	01 c2                	add    edx,eax
  4011d7:	83 e2 01             	and    edx,0x1
  4011da:	29 c2                	sub    edx,eax
  4011dc:	89 55 fc             	mov    DWORD PTR [rbp-0x4],edx
  4011df:	8b 45 fc             	mov    eax,DWORD PTR [rbp-0x4]
  4011e2:	48 98                	cdqe
  4011e4:	48 8d 14 85 00 00 00 	lea    rdx,[rax*4+0x0]
  4011eb:	00 
  4011ec:	48 8d 45 e4          	lea    rax,[rbp-0x1c]
  4011f0:	48 01 d0             	add    rax,rdx
  4011f3:	48 89 c6             	mov    rsi,rax
  4011f6:	bf 40 20 40 00       	mov    edi,0x402040
  4011fb:	b8 00 00 00 00       	mov    eax,0x0
  401200:	e8 4b fe ff ff       	call   401050 <__isoc99_scanf@plt>
  401205:	83 f8 01             	cmp    eax,0x1
  401208:	0f 84 6a ff ff ff    	je     401178 <main+0x32>
  40120e:       83 7d f4 02             cmp    DWORD PTR [rbp-0xc
    ],0x2
  401212:       7f 11                   jg     401225 <main+0xdf>
  401214:       bf 48 20 40 00          mov    edi,0x402048
  401219:       e8 12 fe ff ff          call   401030 <puts@plt>
  40121e:       b8 ff ff ff ff          mov    eax,0xffffffff
  401223:       eb 21                   jmp    401246 <main+0x100>
  401225:       80 7d f3 00             cmp    BYTE PTR [rbp-0xd],0
    x0
  401229:       74 0c                   je     401237 <main+0xf1>
  40122b:       bf 77 20 40 00          mov    edi,0x402077
  401230:       e8 fb fd ff ff          call   401030 <puts@plt>
  401235:       eb 0a                   jmp    401241 <main+0xfb>
  401237:       bf 7b 20 40 00          mov    edi,0x40207b
  40123c:       e8 ef fd ff ff          call   401030 <puts@plt>
  401241:       b8 00 00 00 00          mov    eax,0x0
  401246:       c9                      leave
  401247:       c3                      ret
  401248:       0f 1f 84 00 00 00 00    nop    DWORD PTR [rax+rax
    *1+0x0]
  40124f:       00
```

Here first value 0 is copied to [rbp-0x4] and it used for toggling whenever input comes and then 0 is copied to [rbp-0xc] and it is used to count number of inputs plus along with this we maintain a BYTE PTR [rbp-0xd] in which we move value of 1 first and it will check the difference between two consecutive numbers.Then we jump to "4011df" where we store value of [rbp-0x4] and now address of rdx either becomes "0x0" or adds 4 to it when rax(/eax) is 1. and now address of rax becomes [rbp-1xc] , also the things after till "401200" that are for reading the input numbers. then we compare value in eax(the return value of the scanf@plt) with 1 and if it is equal, then we jump to "401178"
And it "401178" loop the code is just reading the input and no. of inputs and also updating the difference, if that difference isn't same, then that BYTE PTR is setting to zero. In addition we are updating rdx with 4 as in above case with every input (because each integer needs 4 bytes)
Finally if the value of eax is not equa to 1, then we compare first the [rbp-0xc](i.e the no. of inputs) , if that is greater than 2 then jump to "401225", which compares BYTE PTR [rbp-0xd] with 0 and if it is equal, it moves to "401237" and prints "NO" because the difference is not same
Now if BYTE PTR [rbp-0xd] is not equal to 0, then calls $< puts@plt >$ and prints "YES".
Finally if no. of numbers is not greater than 2, then it won't jump to "401225" and prints the prompt "You have not entered enough numbers, try again".
So Conclusion is that the sequence is an AP because it is giving "YES" if the difference is same between all consecutive numbers and "NO" if not.

1.2 Program 2

$< func >$ ***function*** :

```
0000000000401136 <func>:
  401136:       55                      push   rbp
  401137:       48 89 e5                mov    rbp,rsp
  40113a:       53                      push   rbx
  40113b:       48 83 ec 28             sub    rsp,0x28
  40113f:       48 89 7d d8             mov    QWORD PTR [rbp-0x28
    ],rdi
  401143:       48 83 7d d8 00          cmp    QWORD PTR [rbp-0x28
    ],0x0
```

The above code in function is for copying the input number that is stored in "rdi" to address corresponding to [rbp-0x28], and then comparing the value with zero

```
  401148:       75 07                   jne    401151 <func+0x1b>
  40114a:       b8 01 00 00 00          mov    eax,0x1
  40114f:       eb 50                   jmp    4011a1 <func+0x6b>
```

If that value is not equal to zero, then it will jump to "401151" line and if it is equal to zero then value 1 will be moved to register "eax" and then "4011a1" is called after that it will return the value "1"(i.e. $f(0) = 1$)

```
  401151:       48 c7 45 e8 00 00 00    mov    QWORD PTR [rbp-0x18
    ],0x0
  401158:       00
  401159:       48 c7 45 e0 01 00 00    mov    QWORD PTR [rbp-0x20
    ],0x1
  401160:       00
  401161:       eb 30                   jmp    401193 <func+0x5d>

```

After jumping to "401151" in above code, value "0" is copied to [rbp-0x18] and value "1" is copied to [rbp-0x20], and it jump to "401193"

```
  401163:       48 8b 45 e0             mov    rax,QWORD PTR [rbp-0
    x20]
  401167:       48 83 e8 01             sub    rax,0x1
  40116b:       48 89 c7                mov    rdi,rax
  40116e:       e8 c3 ff ff ff          call   401136 <func>
  401173:       48 89 c3                mov    rbx,rax
  401176:       48 8b 45 d8             mov    rax,QWORD PTR [rbp-0
    x28]
  40117a:       48 2b 45 e0             sub    rax,QWORD PTR [rbp-0
    x20]
  40117e:       48 89 c7                mov    rdi,rax
  401181:       e8 b0 ff ff ff          call   401136 <func>
  401186:       48 0f af c3             imul   rax,rbx
  40118a:       48 01 45 e8             add    QWORD PTR [rbp-0x18
    ],rax
  40118e:       48 83 45 e0 01          add    QWORD PTR [rbp-0x20
    ],0x1
  401193:       48 8b 45 e0             mov    rax,QWORD PTR [rbp-0
    x20]
  401197:       48 39 45 d8             cmp    QWORD PTR [rbp-0x28
    ],rax
  40119b:       73 c6                   jae    401163 <func+0x2d>
  40119d:       48 8b 45 e8             mov    rax,QWORD PTR [rbp-0
    x18]
  4011a1:       48 8b 5d f8             mov    rbx,QWORD PTR [rbp-0
    x8]
  4011a5:       c9                      leave
  4011a6:       c3                      ret
```

Here code is performing a loop with each time updating its parameter value. First the value in [rbp-0x20] is copied to rax and value in rax is reduced by 1 and then n(the input) becomes n-1 which now becomes the input and function is again called with input as n-1 and ten when it returns , the rax value is moved to rbx and value in [rbp-0x28] which is the original input is moved to rax and subtract value stored in [rbp-0x20] and that value is stored in ”rdi” and again func is called and then the value which is returned is multiplied by value in ”rbx” and then rax value add with value in [rbp-0x18] and stored in [rbp-0x18] and also value in [rbp-0x20] is incremented by 1 and that value is moved to rax. Again compare that value with original input in [rbp-0x28] and the loop repeats everytime when value at [rbp-0x28] is >= value in rax (that is loop ends when we reach value 0). So the output is in [rbp-0x18] and that is basically this :

$$Output = \sum_{i=0}^{n-1} f(i)f(n-i-1)$$

and finally this value is copied to rax and we return to the main function

$< main >$ ***function*** :

```
00000000004011a7 <main>:
  4011a7:       55                      push   rbp
  4011a8:       48 89 e5                mov    rbp,rsp
  4011ab:       48 83 ec 10             sub    rsp,0x10
  4011af:       bf 08 20 40 00          mov    edi,0x402008
  4011b4:       b8 00 00 00 00          mov    eax,0x0
  4011b9:       e8 72 fe ff ff          call   401030 <printf@plt>

```

This is for displaying the prompt message to take the input ” Enter a non-negative number”

```

  4011be:       48 8d 45 f8             lea    rax,[rbp-0x8]
  4011c2:       48 89 c6                mov    rsi,rax
  4011c5:       bf 27 20 40 00          mov    edi,0x402027
  4011ca:       b8 00 00 00 00          mov    eax,0x0
  4011cf:       e8 6c fe ff ff          call   401040 <
    __isoc99_scanf@plt>
```

This above code is basically for reading the input number that we are giving as input

```
  4011d4:       48 8b 45 f8             mov    rax,QWORD PTR [rbp-0
    x8]
  4011d8:       48 89 c7                mov    rdi,rax
  4011db:       e8 56 ff ff ff          call   401136 <func>
```

In this section, the value corresponding to [rbp-0x8] is copied to "rax" register, and that value is again copied to "rdi" register. Now there is a function call where we will get to know the sequence

```
4011e0:       48 89 c6                  mov    rsi,rax
4011e3:       bf 2c 20 40 00            mov    edi,0x40202c
4011e8:       b8 00 00 00 00            mov    eax,0x0
4011ed:       e8 3e fe ff ff            call   401030 <printf@plt>
4011f2:       b8 00 00 00 00            mov    eax,0x0
4011f7:       c9                        leave
4011f8:       c3                        ret
4011f9:       0f 1f 80 00 00 00 00      nop    DWORD PTR [rax+0x0]
```

Now the address where the output is, is now moved to rsi and finally again a prompt appears showing the output.
So the conclusion is that the sequence is a function which satisfies :

$$f(0) = f(1) = 1$$

and

$$f(x) = \sum_{i=0}^{x-1} f(i)f(x-i-1)$$

which is basically the series of *"Catalan Numbers"*