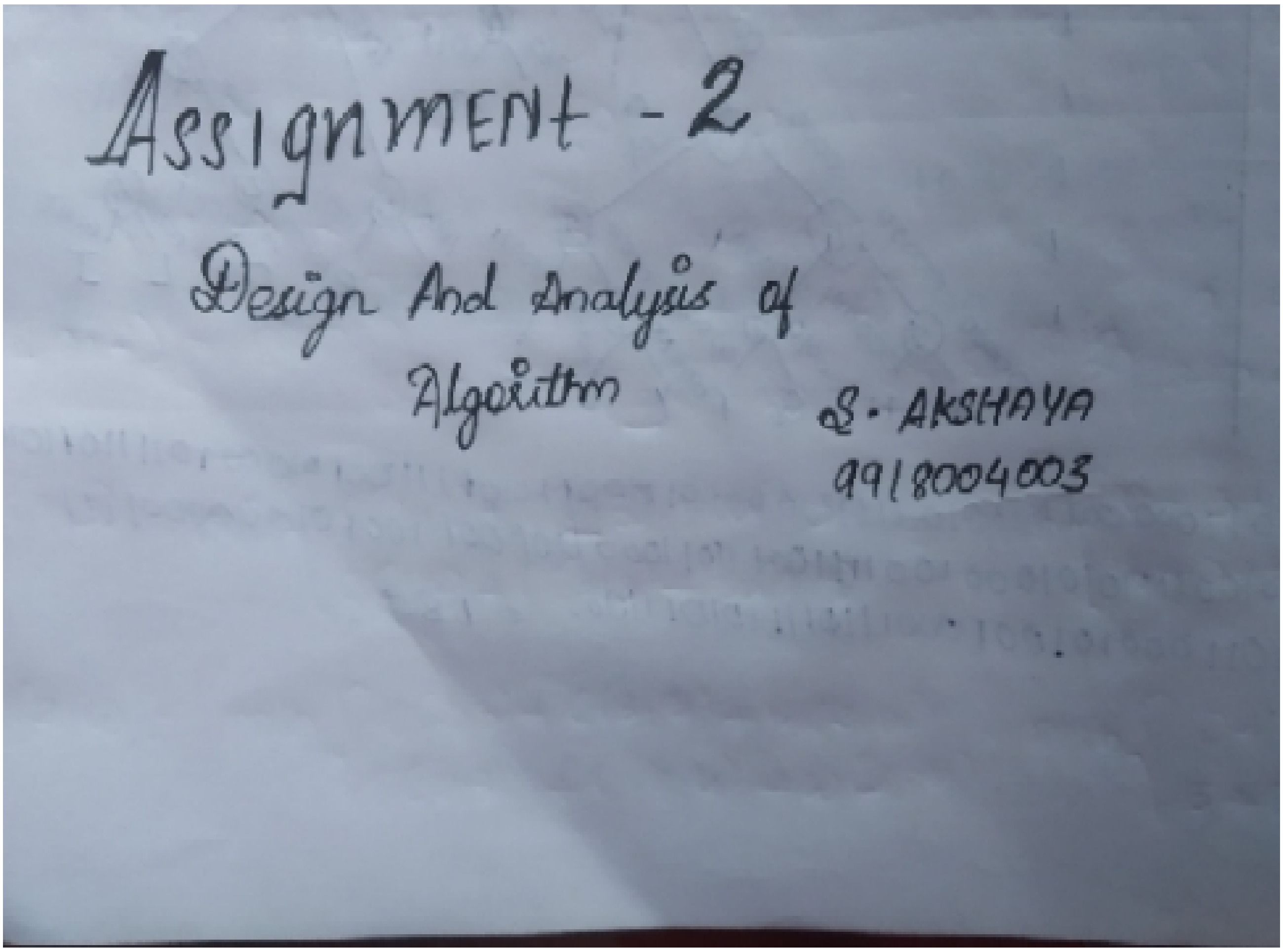


Assignment - 2

Design And Analysis of Algorithm

S. AKSHAYA
9918004003

1) Compress the following string using Huffman coding algorithm

a) AKSHAYA SWAMINATHAN

Number of characters : 19

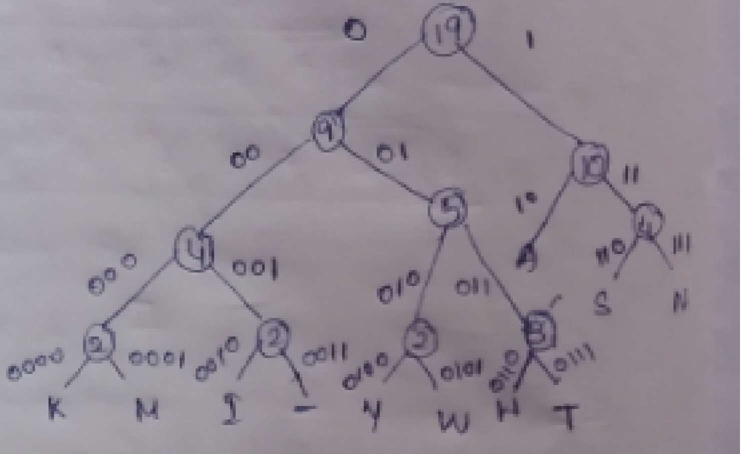
NO of bits required before compression : 19×8 = 152 bits

Frequency of characters:

A B C D E F G H I J K L M

N O P Q R S T U V W X Y Z _

charecta	Frequency
A	6
H	2
N	2
S	2
T	1
W	1
Y	1
_	1
I	1
K	1
M	1
	19



10 0000 110 0110 10 0100 10 0011 110 0110 10 0001 0010 111 10 0111 0110 10 111

NO of bits required after compression is 60 bits.

(b) KALASALINGAM ACADEMY OF RESEARCH AND EDUCATION

no of characters : 46

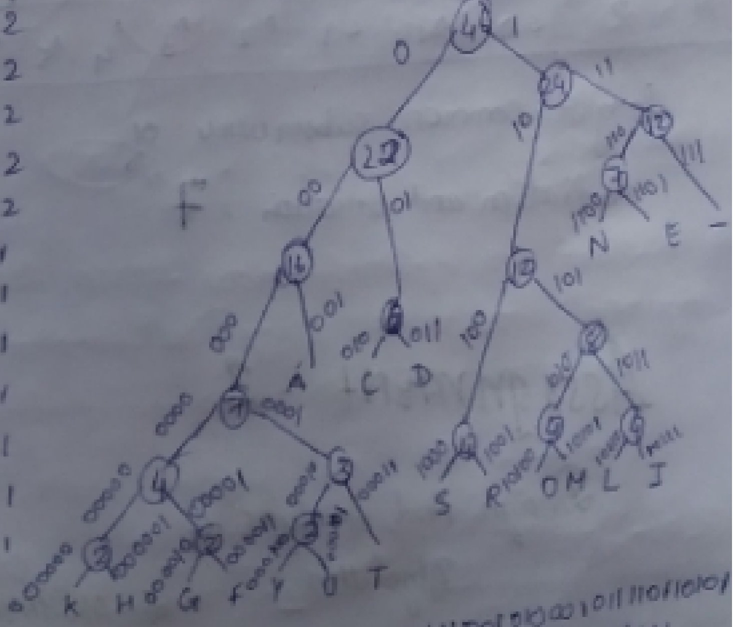
no of bits required before compression : 46×8 : 368 bits

Finding frequency :

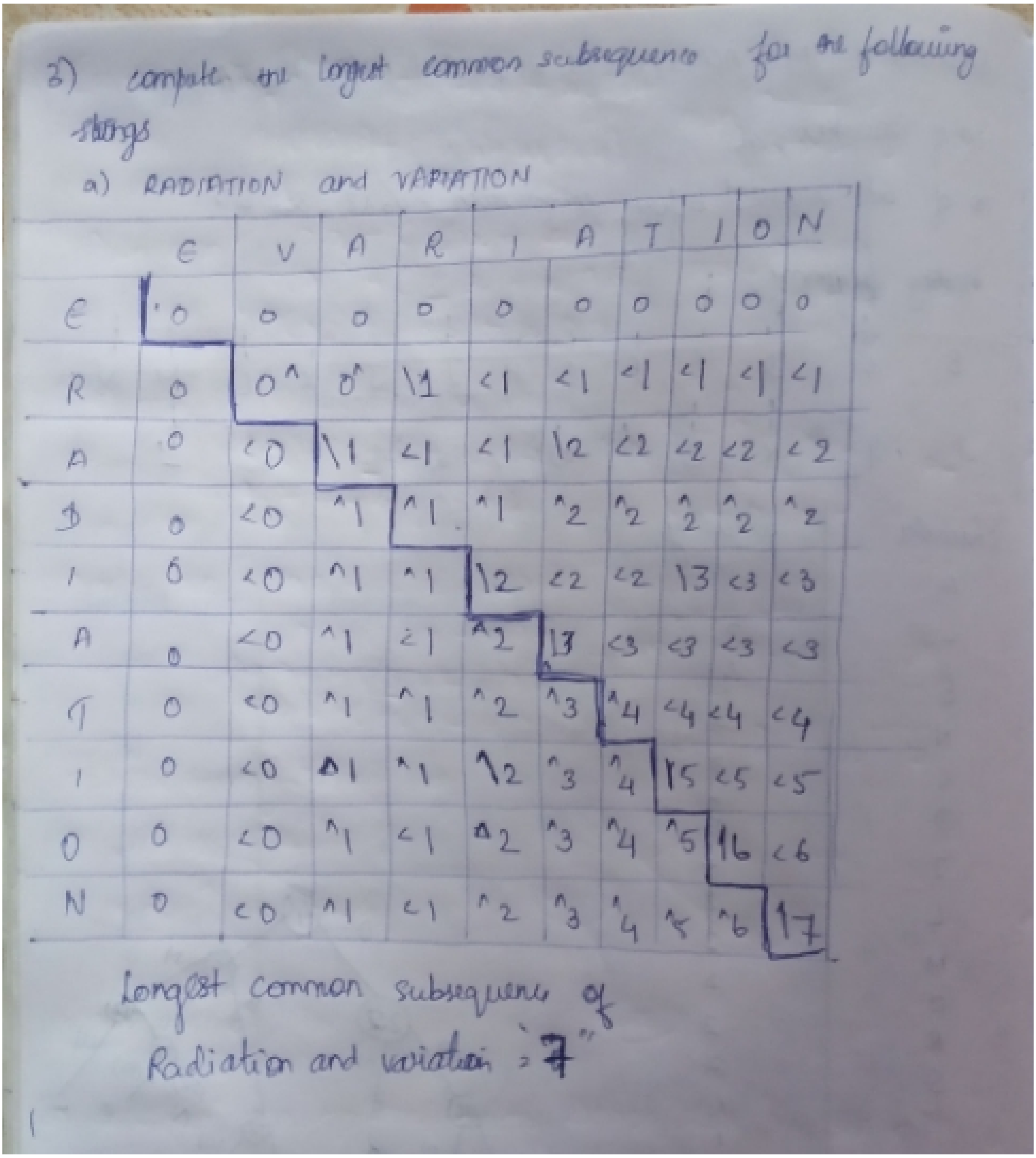
A B C D E F G H I J K L M N

O P Q R S T U V W X Y Z _

character	frequency
A	9
_	5
E	4
N	3
C	3
D	3
I	2
L	2
M	2
O	2
R	2
S	2
T	1
U	1
Y	1
F	1
G	1
H	1
K	1



0000000110110 0011000001 011010 11111 00 00001 0001 1010 1 111 001 010 00 1011 1101 1010 1
000100111 10101 000100111 10101000 10011 1001 1101 1000 1101 001 1001 0100 0000 1111
001 1100011111 110 1011 00010 1001 0001 1111 1 10101 1100. = 157.



3) compute the longest common subsequence for the following strings

a) RADIATION and VARIATION

	E	V	A	R	I	A	T	I	O	N
E	0	0	0	0	0	0	0	0	0	0
R	0	0^	0^	\1	<1	<1	<1	<1	<1	<1
A	0	<0	\1	<1	<1	\2	<2	<2	<2	<2
D	0	<0	^1	^1	^1	^2	^2	^2	^2	^2
I	0	<0	^1	^1	\2	<2	<2	\3	<3	<3
A	0	<0	^1	<1	^2	\3	<3	<3	<3	<3
T	0	<0	^1	^1	^2	^3	^4	<4	<4	<4
I	0	<0	^1	^1	\2	^3	^4	\5	<5	<5
O	0	<0	^1	<1	^2	^3	^4	^5	\6	<6
N	0	<0	^1	<1	^2	^3	^4	^5	^6	\7

Longest common subsequence of Radiation and variation : "7"

b) ALGORITHMS

	E	L	O	G	A	R	I	T	H	M	S
E	0	0	0	0	0	0	0	0	0	0	0
A	0	<0	<0	<0	\1	<1	<1	<1	<1	<1	<1
L	0	\1	<1	<1	\1	<1	<1	<1	<1	<1	<1
G	0	^1	<1	\2	<2	<2	<2	<2	<2	<2	<2
O	0	^1	\2	<2	<2	<2	<2	<2	<2	<2	<2
R	0	^1	^2	<2	<2	\3	<3	<3	<3	<3	<3
I	0	^1	^2	<2	<2	^3	\4	<4	<4	<4	<4
T	0	^1	^2	<2	<2	^3	^4	\5	<5	<5	<5
H	0	^1	^2	<2	<2	^3	^4	^5	\6	<6	<6
M	0	^1	^2	<2	<2	^3	^4	^5	^6	\7	<7
S	0	^1	^2	^2	^2	^3	^4	^5	^6	^7	\8

Longest common subsequence of

Logarithms and algorithm is 8.

3) construct an optional binary search tree for the following keys and corresponding frequence : [key:freq]

keys → Rank Name Email Regno

frequency → 5 3 6 7

i \ j	0	1	2	3	4
0	0	5	$11^{(1)}$		
1		0	3	$12^{(3)}$	
2			0	6	
3				0	7
4					0

$$C(0,2) = \begin{cases} C(0,0) + C(\underline{1},2) + w(0,2) \\ C(0,1) + (2,2) + w(0,2) \end{cases}$$

$$= \min \begin{cases} 0+3+8 = 11 \\ 5+0+5 = 10 \end{cases} = 11^{(1)}$$

$$C(1,3) = \min \begin{cases} C(1,1) + C(2,3) + w(1,3) \\ C(1,2) + C(\underline{3},3) + w(1,3) \end{cases}$$

$$= \min \begin{cases} 0+6+9 = 12^{(3)} \\ 3+0+9 \end{cases}$$

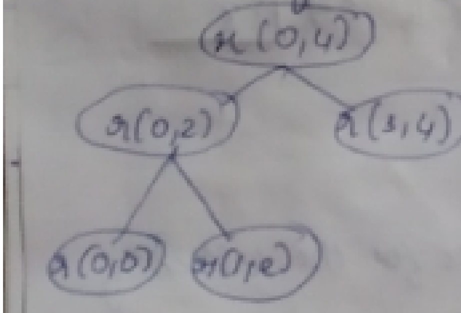
$$c(2,4) = \min\begin{cases} C(2,2)+C(3,4)+w(2,4) \\ C(2,3)+C(\underline{4},4)+w(2,4) \end{cases} = \min\begin{cases} 0+7+13 \\ 6+0+13 \end{cases} = 19^{(4)}$$

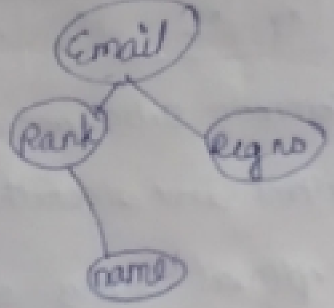
$$c(0,3) = \min\begin{cases} C(0,0)+C(1,3)+w(0,3) \\ C(0,1)+C(\underline{2},3)+w(0,3) \\ C(0,2)+C(3,3)+w(0,3) \end{cases} = \min\begin{cases} 0+12+14 \\ 5+6+14 \\ 11+0+14 \end{cases} = 25(2)$$

$$c(1,4) = \min\begin{cases} C(1,1), C(2,4)+w(1,4) \\ C(1,2)+C(\underline{3},4)+w(1,4) \\ C(1,3)+C(4,4)+w(1,4) \end{cases} = \min\begin{cases} 0+19+16 \\ 3+7+16 \\ 25+0+16 \end{cases} = 3(4)$$

$$c(0,4) = \min\begin{cases} C(0,0)+C(1,4)+w(0,4) \\ C(0,1)+C(2,4)+w(0,4) \\ C(0,2)+C(\underline{3},4)+w(0,4) \\ C(0,3)+C(4,4)+w(0,4) \end{cases} = \min\begin{cases} 0+26+21 \\ 5+19+21 \\ 11+7+21 \\ 25+0+21 \end{cases} = 39^{(3)}$$

optimal binary search tree:





④ construct an optimal binary search tree for the following keys and corresponding frequencies denoted an {key, freq}

key → 10 20 30 40

freq → 2 6 3 4

$$C(0,2) = \min\begin{cases} C(0,0)+(1,2)+w(0,2) \\ C(0,1)+C(\underline{2},2)+w(0,2) \end{cases} = \min\begin{cases} 0+6+8 \\ 2+0+8 \end{cases} = 10^{(2)}$$

$$c(1,3) = \min\begin{cases} C(1,1)+C(\underline{2},3)+w(1,3) \\ C(1,2)+C(3,3)+w(1,3) \end{cases} = \min\begin{cases} 0+3+9 \\ 6+0+9 \end{cases} = 12^{(2)}$$

$$c(2,4) = \min\begin{cases} C(2,3)+C(3,4)+w(2,4) \\ C(2,3)+C(4,4)+w(2,4) \end{cases} \begin{cases} 0+4+7 \\ 3+0+7 \end{cases}$$

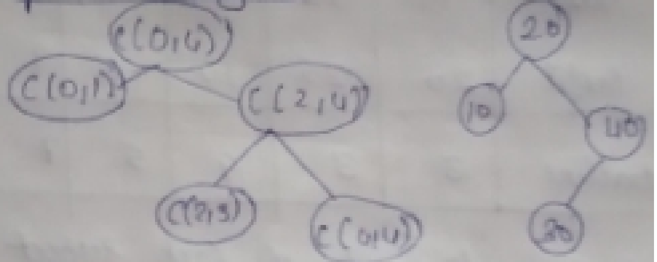
$$c(0,3) = \min\begin{cases} c(0,0) + c(1,3) + w(0,3) \\ c(0,1) + c(\underline{2},3) + w(0,3) \\ c(0,2) + c(3,3) + w(0,3) \end{cases} = \min\begin{cases} 0+12+11 \\ 2+3+11 \\ 10+0+11 \end{cases} = 16^{(2)}$$

$$c(1,4) = \min\begin{cases} c(1,1) + c(\underline{2},4) + w(1,4) \\ c(1,2) + c(3,4) + w(1,4) \\ c(1,3) + c(4,4) + w(1,4) \end{cases} = \min\begin{cases} 0+10+13 \\ 10+4+13 \\ 16+0+13 \end{cases} = 23^{(2)}$$

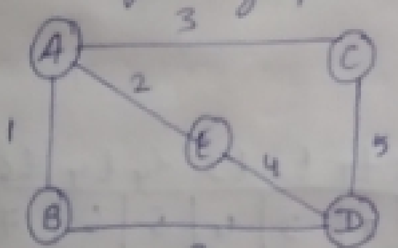
$$c(0,4) = \min\begin{cases} c(0,0) + c(1,4) + w(0,4) \\ c(0,1) + c(2,4) + w(0,4) \\ c(0,2) + c(3,4) + w(0,4) \\ c(0,3) + c(4,4) + w(0,4) \end{cases} = \min\begin{cases} 0+23+15 \\ 2+10+15 \\ 10+4+15 \\ 16+0+15 \end{cases} = 27^{(2)}$$

j \ i	0	1	2	3	4
0	0	2	10^2	16^2	27^2
1		0	6	12^2	23^2
2			0	3	10^4
3				0	4
4					0

optimal binary tre



5) find the shotest pattan from node A to all the other nodes & the given graph



Source	A	B	C	D	E	choosen vertex
init{}	0	∞	∞	∞	∞	A
{A}	–	1,A	3,A	∞	2,A	B
{A,B}	–	–	3,A	2,B 2,A	2,A	E
{A,E}	–	1,A	3,A	2,B 3,A	–	C
{A,C}	-	1,A	–	2,B 3,A	-	D
{A,B,D}	-	1,A	3,A	–	2,A	-

Result of shortest pattern from A :

Source	Destination	pattern	cost
A	B	AB	1
A	C	AC	3
A	D	ABD	3
A	E	AE	2

6. Schedule the following tasks using job scheduling algorithm. Assume that each task executes in unit time and no two tasks can execute at the same time

Tasks	T1	T2	T3	T4	T5	T6	T7	T8	T9
Profit	20	25	10	15	9	22	19	22	30
deadline	3	3	4	5	7	7	6	2	2

Sol Descending order of task respect to profit

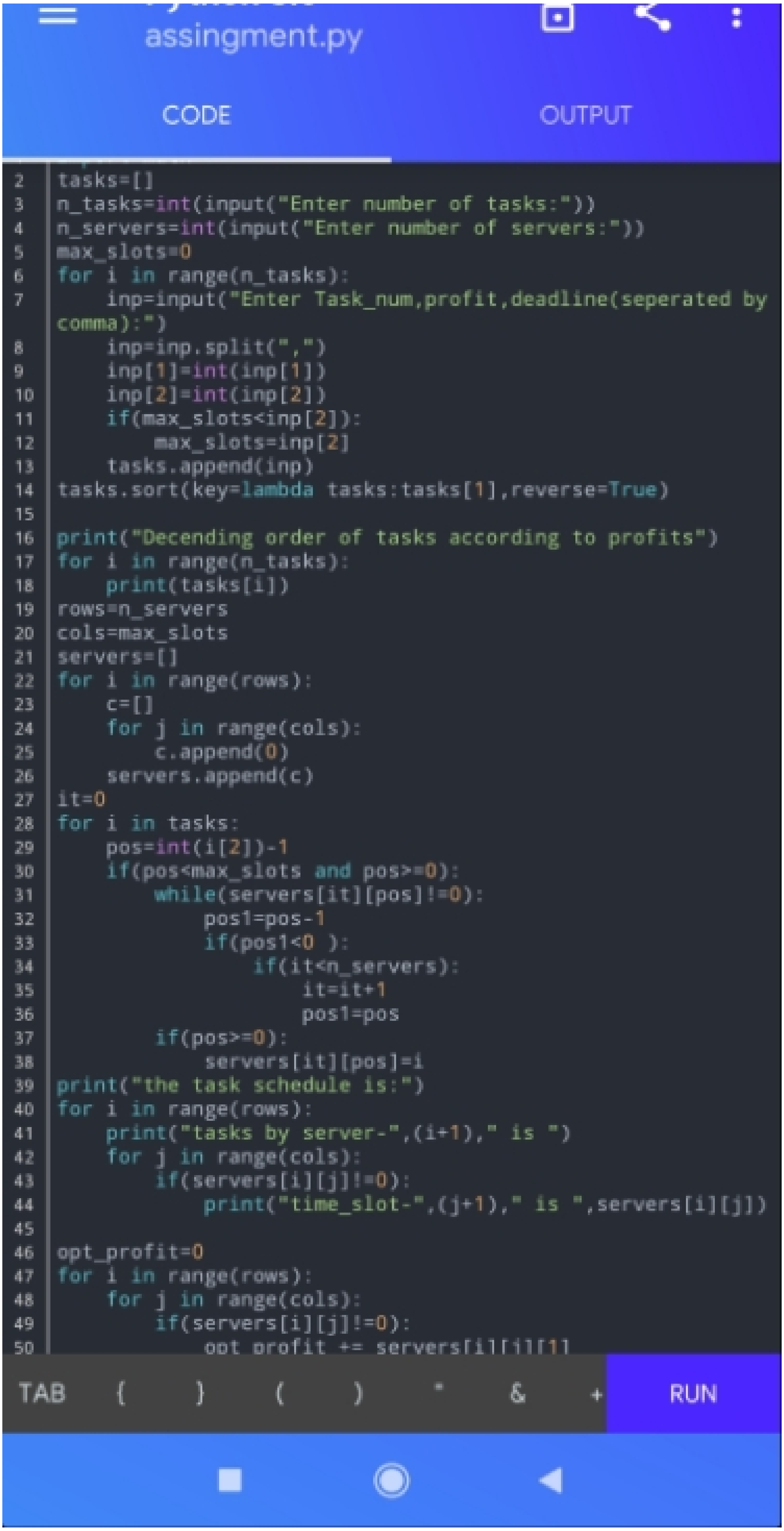
Task	profit	deadline
T9	30	2
T2	25	3
T6	22	7
T8	22	2
T1	20	3
T7	19	6
T4	15	5
T3	10	4
T5	9	7

t8	t9	t2	t3	t4	t7	t6
1	2	3	4	5	6	7

optimal schedule : T8, t9, t2, t3, t4, t7t6

max profit : 22 + 30 + 25 + 10 + 15 + 19 +

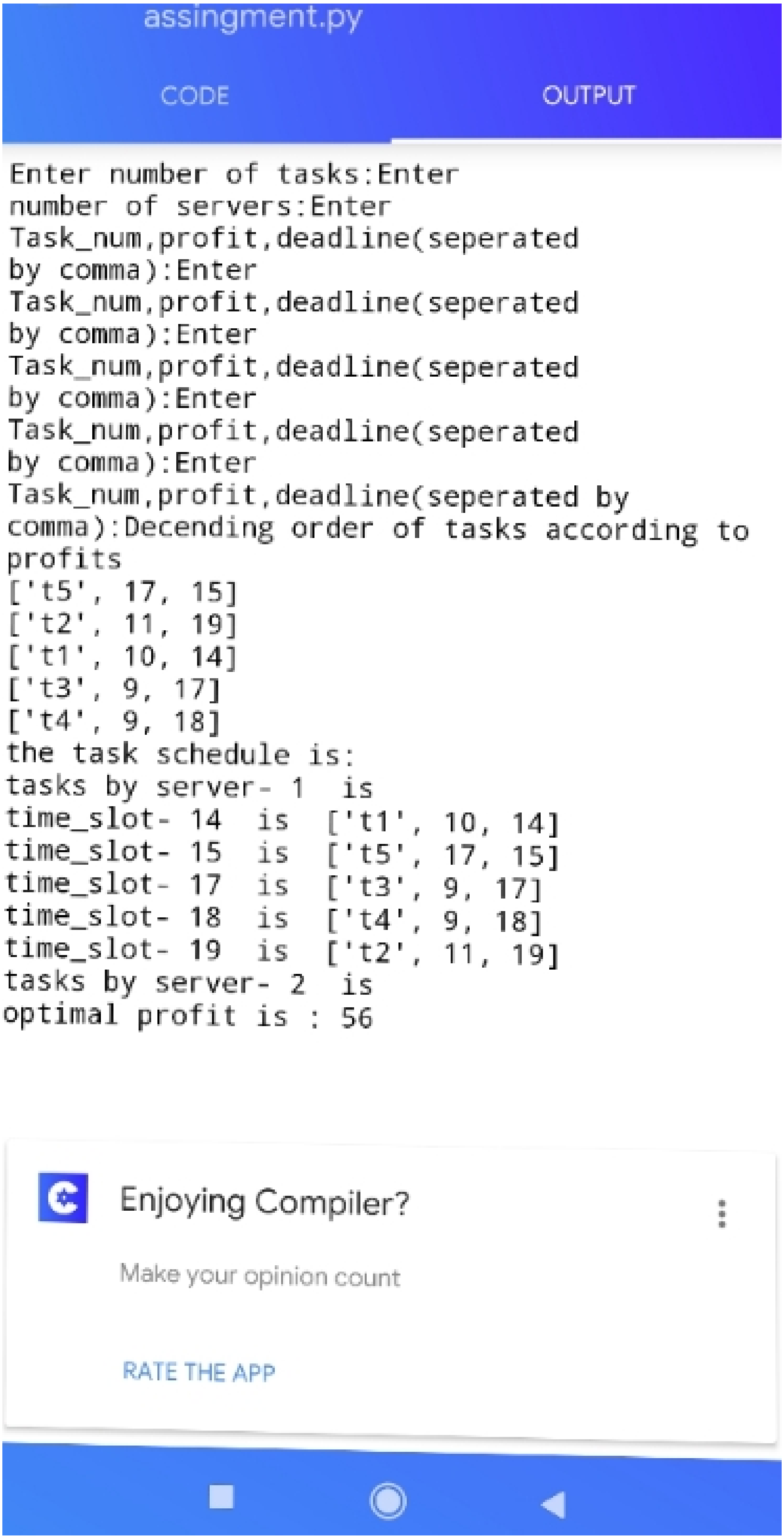
= 143



```
tasks=[]
n_tasks=int(input("Enter number of tasks:"))
n_servers=int(input("Enter number of servers:"))
max_slots=0
for i in range(n_tasks):
    inp=input("Enter Task_num,profit,deadline(seperated by comma):")
    inp=inp.split(",")
    inp[1]=int(inp[1])
    inp[2]=int(inp[2])
    if(max_slots<inp[2]):
        max_slots=inp[2]
    tasks.append(inp)
tasks.sort(key=lambda tasks:tasks[1],reverse=True)

print("Decending order of tasks according to profits")
for i in range(n_tasks):
    print(tasks[i])
rows=n_servers
cols=max_slots
servers=[]
for i in range(rows):
    c=[]
    for j in range(cols):
        c.append(0)
    servers.append(c)
it=0
for i in tasks:
    pos=int(i[2])-1
    if(pos<max_slots and pos>=0):
        while(servers[it][pos]!=0):
            pos1=pos-1
            if(pos1<0 ):
                if(it<n_servers):
                    it=it+1
                    pos1=pos
        if(pos>=0):
            servers[it][pos]=i
print("the task schedule is:")
for i in range(rows):
    print("tasks by server-",(i+1)," is ")
    for j in range(cols):
        if(servers[i][j]!=0):
            print("time_slot-",(j+1)," is ",servers[i][j])

opt_profit=0
for i in range(rows):
    for j in range(cols):
        if(servers[i][j]!=0):
            opt_profit += servers[i][j][1]
```



```
Enter number of tasks:Enter
number of servers:Enter
Task_num,profit,deadline(seperated
by comma):Enter
Task_num,profit,deadline(seperated
by comma):Enter
Task_num,profit,deadline(seperated
by comma):Enter
Task_num,profit,deadline(seperated
by comma):Enter
Task_num,profit,deadline(seperated by
comma):Decending order of tasks according to
profits
['t5', 17, 15]
['t2', 11, 19]
['t1', 10, 14]
['t3', 9, 17]
['t4', 9, 18]
the task schedule is:
tasks by server- 1  is
time_slot- 14  is  ['t1', 10, 14]
time_slot- 15  is  ['t5', 17, 15]
time_slot- 17  is  ['t3', 9, 17]
time_slot- 18  is  ['t4', 9, 18]
time_slot- 19  is  ['t2', 11, 19]
tasks by server- 2  is
optimal profit is : 56
```