

PROGRAM 1:

```
echo "To identify the current shell and length of the string"
```

```
echo "The current shell is=$SHELL"
```

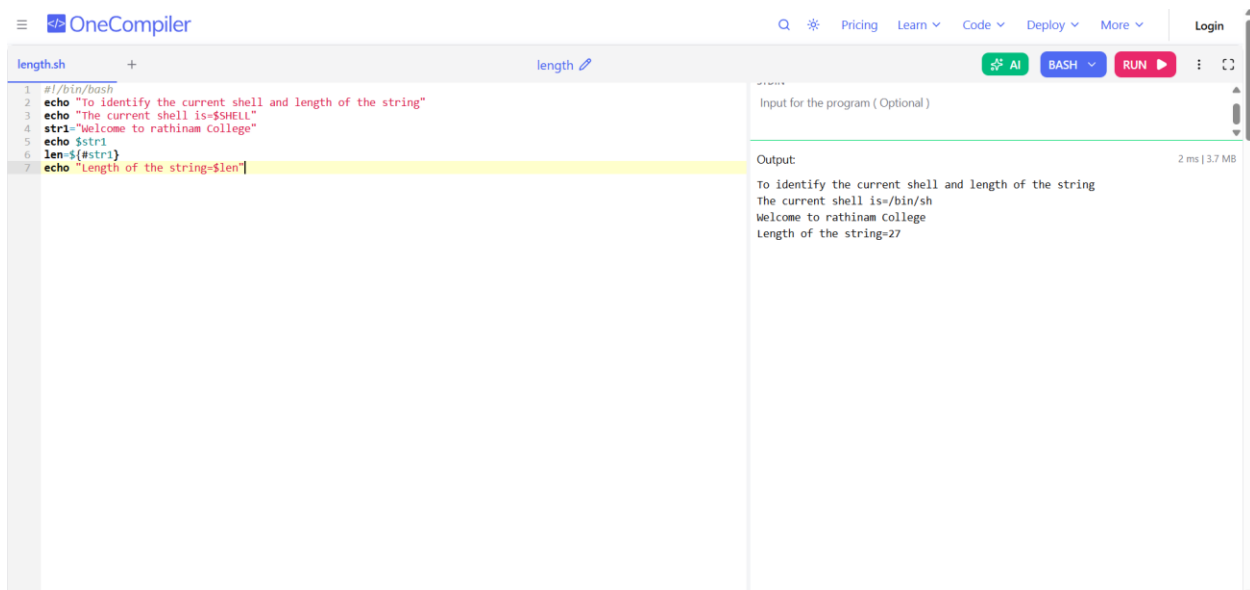
```
str1="Welcome to rathinam College"
```

```
echo $str1
```

```
len=${#str1}
```

```
echo "Length of the string=$len"
```

OUTPUT:



The screenshot shows the OneCompiler online IDE interface. The code editor on the left contains a shell script named 'length.sh' with the following content:

```
1 #!/bin/bash
2 echo "To identify the current shell and length of the string"
3 echo "The current shell is=$SHELL"
4 str1="Welcome to rathinam College"
5 echo $str1
6 len=${#str1}
7 echo "Length of the string=$len"
```

The right-hand side of the interface shows the output of the script. It includes a section for 'Input for the program (Optional)' and an 'Output:' section. The output text is:

```
To identify the current shell and length of the string
The current shell is=/bin/sh
Welcome to rathinam College
Length of the string=27
```

At the bottom right of the output section, the execution time and memory usage are displayed as '2 ms | 3.7 MB'.

PROGRAM 2:

```
echo "-----"
```

```
echo "To count backwards for 100 to 0 using for loop"
```

```
echo "-----"
```

```
c=0
```

```
for ((i=100; i>=0; --i))
```

```
do
```

```
    echo "$i"
```

```

s=${c++}

done

echo "-----"

echo "Count=$s"

echo "-----"

```

OUTPUT:

The screenshot shows a terminal window with a script being executed. The script is as follows:

```

1 #!/bin/bash
2 echo "-----"
3 echo "To count backwards for 100 to 0 using for loop"
4 echo "-----"
5 c=0
6 for ((i=100; i>=0; --i))
7 do
8     echo "$i"
9     s=${c++}
10 done
11 echo "-----"
12 echo "Count=$s"
13 echo "-----"

```

The output of the script is shown on the right side of the terminal window, displaying a list of numbers from 100 down to 0, followed by the final count:

```

24
23
22
21
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
0
-----
Count=100
-----

```

PROGRAM 3:

```

echo "-----"

echo "Odd and Even numbers upto 100"

echo "-----"

even=0

odd=0

sum_of_odd=0

sum_of_even=0

```

```

for ((i=0; i<=100; i++))
do

    if (( i % 2 == 0 ))

    then

        ((even++))

        ((sum_of_even += i))

    else

        ((odd++))

        ((sum_of_odd += i))

    fi

done

echo "-----"

echo "Even Count   = $even"

echo "Odd Count    = $odd"

echo "Sum of Evens  = $sum_of_even"

echo "Sum of Odds   = $sum_of_odd"

echo "-----"

```

The screenshot shows a terminal window with a shell script being executed. The script is named 'script1.sh' and is located in the directory '44cphvsn'. The script's output is displayed on the right side of the terminal window.

```

script1.sh  +  44cphvsn
1 #!/bin/bash
2 echo "-----"
3 echo "Odd and Even numbers upto 100"
4 echo "-----"
5 even=0
6 odd=0
7 sum_of_even=0
8 sum_of_odd=0
9 for ((i=0; i<=100; i++))
10 do
11     if (( i % 2 == 0 ))
12     then
13         ((even++))
14         ((sum_of_even += i))
15     else
16         ((odd++))
17         ((sum_of_odd += i))
18     fi
19 done
20 echo "-----"
21 echo "Even Count   = $even"
22 echo "Odd Count    = $odd"
23 echo "Sum of Evens  = $sum_of_even"
24 echo "Sum of Odds   = $sum_of_odd"
25 echo "-----"

```

STDIN
Input for the program (Optional)

Output: 2 ms | 3.9 MB

```

-----
Odd and Even numbers upto 100
-----
Even Count   = 51
Odd Count    = 50
Sum of Evens = 2550
Sum of Odds  = 2500
-----

```

PROGRAM 4

```
echo "-----"

echo "To print the star in the descending order"

echo "-----"

for ((i=5; i>=1; i--))
do

    for ((j=1; j<=i; j++))

    do

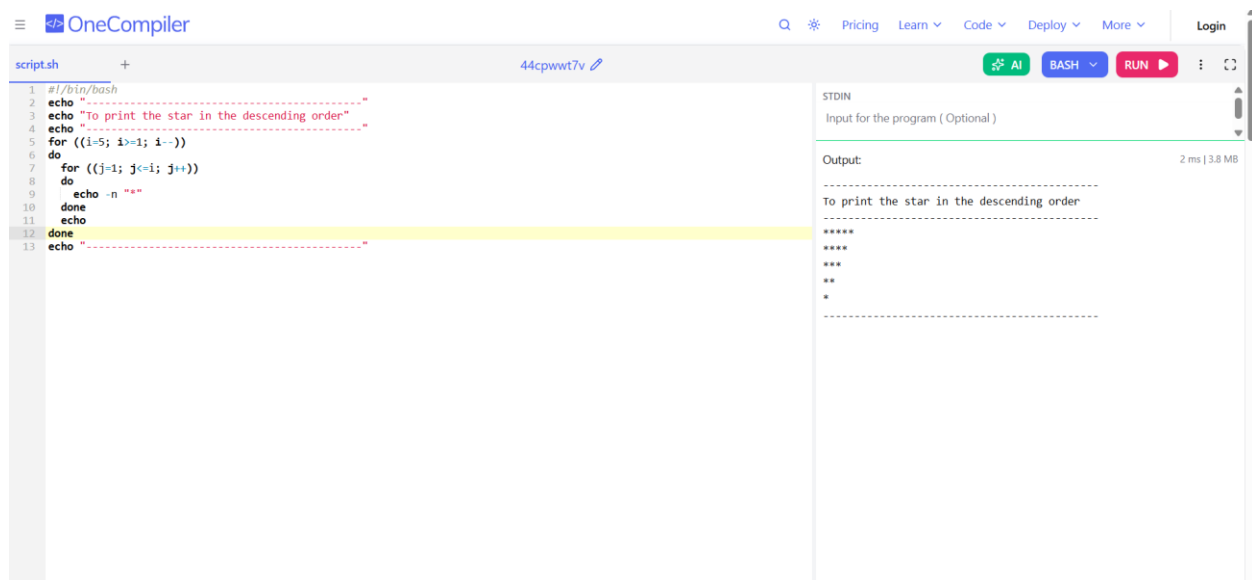
        echo -n "*"

    done

    echo

done

echo "-----"
```



The screenshot shows the OneCompiler online IDE interface. The code editor on the left contains a Bash script with 13 lines. The script prints a dashed line, a message, another dashed line, a descending star pattern, and a final dashed line. The output panel on the right shows the execution results, including the message and the star pattern. The star pattern consists of five lines: the first line has five stars, the second has four, the third has three, the fourth has two, and the fifth has one. The output panel also shows the execution time as 2 ms and memory usage as 3.8 MB.

```
1 #!/bin/bash
2 echo "-----"
3 echo "To print the star in the descending order"
4 echo "-----"
5 for ((i=5; i>=1; i--))
6 do
7     for ((j=1; j<=i; j++))
8     do
9         echo -n "*"
10    done
11    echo
12 done
13 echo "-----"
```

STDIN
Input for the program (Optional)

Output: 2 ms | 3.8 MB

```
-----
To print the star in the descending order
-----
*****
****
***
**
*
-----
```