1st WEEK

Name: B Lohith Krishnan

Roll no:CH.SC.U4CSE24153

Program 1:

Code:

```
#include <stdio.h>

int sumoffirstn(int n) {
n=(n*(n+1))/2;
return n;
}

int main() {
int n,sum;
printf("Enter a num: ");
scanf("%d",&n);
sum=sumoffirstn(n);
printf("The sum of first %d numbers is: %d\n",n,sum);
}
```

Output:

```
C:\DAA>gcc q1.c

C:\DAA>a
Enter a num: 5
The sum of first 5 numbers is: 15
```

Space-Complexity Justification:

Fixed part:

Variables: n,sum,n(in the function)

Space used:(1+1+1)*4=3*4=12

Variable part:

No array, recursion or any data structure used, So

Space used: 0

Space = 12= constant

Space complexity =O(1)

Program 2:

Code:

```
#include <stdio.h>

int main() {
int n,sum=0;
printf("Enter a num: ");
scanf("%d",&n);
for(int i=1;i<=n;i++) {
sum=sum+(i*i);
}
printf("The sum of squares of first %d numbers is: %d\n",n,sum);
}
```

Output:

```
C:\DAA>gcc q2.c

C:\DAA>a
Enter a num: 6
The sum of squares of first 6 numbers is: 91
```

Space-Complexity Justification:

Fixed part:

Variables: n,sum,i

Space used:(1+1+1)*4=3*4=12

Variable part:

No array, recursion or any data structure used, So

Space used: 0

Space = 12= constant

Space complexity =O(1)

Program 3:

Code:

```
#include <stdio.h>

int main() {
int n,sum=0;
printf("Enter a num: ");
scanf("%d",&n);
for(int i=1;i<=n;i++) {
sum=sum+(i*i*i);
}
printf("The sum of cubes of first %d numbers is: %d\n",n,sum);
}
```

Output:

```
C:\DAA>gcc q3.c

C:\DAA>a
Enter a num: 6
The sum of cubes of first 6 numbers is: 441
```

Space-Complexity Justification:

Fixed part:

Variables: n,sum,i

Space used:(1+1+1)*4=3*4=12

Variable part:

No array, recursion or any data structure used, So

Space used: 0

Space = 12= constant

Space complexity =O(1)

Program 4:

Code:

```
#include <stdio.h>

int fact(int n) {
if(n==1 || n==0) {
return 1;
}
else{
return (n*fact(n-1));
}
}

int main() {
int n;
printf("Enter a num: ");
scanf("%d",&n);
printf("The factorial of %d is: %d\n",n,fact(n));
}
```

Output:

```
C:\DAA>gcc q4.c

C:\DAA>a
Enter a num: 5
The factorial of 5 is: 120
```

Space-Complexity Justification:

Fixed part:

Variables: n,n(in the function)

Space used:(1+1)*4=2*4=8

Variable part:

Fact(n) calls Fact(n-1) until n becomes 0 or 1

Recursion for n times, So

Space used: n

Space = 8+n

Space complexity =O(n)

Program 5:

Code:

```
#include <stdio.h>
#define r 3
#define c 3
int main() {
int mat[r][c]= {{1,2,3},{4,5,6},{7,8,9}};
int tmat[r][c];

for(int i=0;i<r;i++) {
for(int j=0;j<c;j++) {
tmat[i][j]=mat[j][i];
}
}

for(int i=0;i<r;i++) {
for(int j=0;j<c;j++) {
printf("%d ",tmat[i][j]);
}
printf("\n");
}
}
```

Output:

```
C:\DAA>gcc q5.c

C:\DAA>a
1 4 7
2 5 8
3 6 9
```

Space-Complexity Justification:

Fixed part:

Variables: mat[3][3], tmat[3][3], i, j, r, c

Space used:(3*3+3*3+1+1+1+1)*4=22*4=88

Variable part:

Array used in the program doesn’t depend on the input n, So

Space used: 0

Space = 88= constant

Space complexity =O(1)

Program 6:

Code:

```c
#include <stdio.h>

int fibo(int n) {
if(n==1){
return 0;
}
else if(n==2){
return 1;
}
else {
return fibo(n-1)+fibo(n-2);
}
}

int main() {
int n;
printf("Enter a num: ");
scanf("%d",&n);
for(int i=1;i<=n;i++) {
printf("%d ", fibo(i));
}
}
```

Output:

```
C:\DAA>gcc q6.c

C:\DAA>a
Enter a num: 9
0 1 1 2 3 5 8 13 21
```

Space-Complexity Justification:

Fixed part:

Variables: n, i, n(in the function)

Space used:(1+1+1)*4=3*4=12

Variable part:

The function calls itself recursively n times , So

Space used: n

Space = 12 + n

Space complexity =O(n)