Q1. Define a function to input variable length string and store it in an array without memory wastage.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

char *input() {

        char ch;
        char *p, *q;
        int len = 1, i;

        p = (char *)malloc(len);

        while ((ch = getche()) != 13) {

                p[len - 1] = ch;
                q = (char *)malloc(len + 1);
                for (i = 0; i < len; i++)
                        q[i] = p[i];
                q[i] = '\0';

                free(p);
                len++;

                p = (char *)malloc(len);
```

```
            for (i = 0; i < len; i++)
                p[i] = q[i];

            free(q);
        }
        return p;
}

int main() {
        char *z = input();
        printf("your text -> %s", z);
}
```

Q2. Write a program to ask the user to input a number of data values he would like to enter then create an array dynamically to accommodate the data values. Now take the input from the user and display the average of data values

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

int input() {

	int total_num = 2, sum = 0;

	printf("Enter the total input you want to enter -> ");
	scanf("%d", &total_num);

	int *num_arr = (int *)malloc(sizeof(int) * total_num);

	for (int i = 0; i < total_num; i++) {
		printf("%d)= ", i + 1);
		scanf("%d", &num_arr[i]);

		sum = sum + num_arr[i];
	}

	return (sum / total_num);
}
```

```
int main() {
	int avg = input();
	printf("Average = %d", avg);
}
```

Q3. Write a program to calculate the sum of n numbers entered by the user using malloc and free.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

int main() {

	int num, *arr;
	int sum = 0;

	printf("How many times you enterd number = ");
	scanf("%d", &num);

	arr = (int *)malloc(sizeof(int) * num);

	for (int i = 0; i < num; i++) {
		printf("%d) = ", i + 1);
		scanf("%d", &arr[i]);

		sum = sum + arr[i];
	}
	free(arr);
	printf("Sum = %d", sum);

}
```

Q4. Write a program to input and print text using dynamic memory allocation.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

int main() {

	char *name;

	name = (char *)malloc(sizeof(char) * 200);

	fgets(name, 200, stdin);

	printf("%s", name);

	free(name);

}
```

Q5. Write a program to read a one dimensional array, print sum of all elements along with inputted array elements using dynamic memory allocation.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

int main() {

	int arr[10] = {45, 12, 80, 42, 36, 94, 28, 59, 43, 19};
	int *drr = (int *)malloc(sizeof(int));
	int i = 0, sum = 0;

	do {
		drr[i] = arr[i];

		int *tmp = (int *)malloc(sizeof(int) * (i + 1));
		for (int j = 0; j <= i; j++)
			tmp[j] = drr[j];

		free(drr);
		i++;
		drr = (int *)malloc(sizeof(int) * (i + 1));

		for (int j = 0; j <= (i - 1); j++) {
			drr[j] = tmp[j];
		}
```

```
            free(tmp);

        } while (i < 10);

        printf("Inputted array : ");
        for (int j = 0; j < 10; j++) {
            printf("%d ", drr[j]);

            sum = sum + drr[j];
        }

        free(drr);

        printf("\n\nSum of all elements : %d", sum);

}
```

Q6. Write a program in C to find the largest element using Dynamic Memory Allocation.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

int main() {

	int size, larg = 0;

	printf("How many elements you want to enter: ");
	scanf("%d", &size);
	int *arr = (int *)calloc(size, sizeof(int));

	for (int i = 0; i < size; i++) {
		printf("%d) ", i + 1);
		scanf("%d", &arr[i]);

		if (arr[larg] < arr[i]) {
			larg = i;
		}
	}

	printf("Largest element : %d", arr[larg]);
	free(arr);
}
```

Q7. Write a program to demonstrate memory leak in C..

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

void Demo(void) {
        int *ptr = (int *)malloc(sizeof(int));

        int num = 100;

        ptr = &num;
}

int main() {

        Demo();

        printf("I am outside the function\n\n");
        printf("Pointer ptr is not free thats why pointer ptr is in the memory and ptr creates memory leak");
}
```

Q8. Write a program to demonstrate dangling pointers in C.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>

int *Demo(void) {
        int a = 5;
        return &a;
}

int main() {

        int *p = Demo();

}
```

Q9. Write a program to allocate memory dynamically of the size in bytes entered by the user. Also handle the case when memory allocation is failed.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>
int main() {

	int byte;

	printf("How much bytes you want to enter -> ");
	scanf("%d", &byte);

	char *str = (char *)malloc(byte + 1);

	if (str == NULL) {
		printf("Error : memory not allocate");
	} else {

		printf("Enter name : ");

		for (int i = 0; i < byte; i++)
			str[i] = getche();
		str[byte] = '\0';
		printf("\n%s", str);
	}
}
```

Q10. Find out the maximum and minimum from an array using dynamic memory allocation in C.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>


int main() {
	int size, small = 0, large = 0;

	printf("How much number you want to enter : ");
	scanf("%d", &size);

	int *arr = (int *)malloc(sizeof(int) * size);

	for (int i = 0; i < size; i++) {

		printf("%d) ", i);
		scanf("%d", &arr[i]);

		if (large < arr[i])
			large = arr[i];
		else
			small = arr[i];
	}
	printf("largest = %d  and  smallest = %d", large, small);
}
```