

Name: Yashwant Chandrakant Bhosale
MIS: 612303039
DIV: SY COMP DIV 1

Q1. Write a function to count number of occurrences of a character in a string code:

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

int count_occurrence(char str[], char c) {
    int count = 0;
    for(int i = 0; str[i] != '\0'; i++) {
        if(str[i] == c)
            count++;
    }
    return count;
}

int main() {
    char string[64];
    strcpy(string, "hello");
    printf("Occurrence of 'l' in %s: %d\n", string,
count_occurrence(string, 'l'));
    return 0;
}
```

output:

```
Occurrence of 'l' in hello: 2
```

Q2. Write the strtok() function.

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

char *str_tok(char *str, char *delim) {
    static char *ptr = NULL;
    if (str) ptr = str;
    if (ptr == NULL) return NULL;
    char *start = ptr;
    char *p = ptr;
    while (*p != '\0' && *p != *delim) p++;
    if (*p == '\0') {
        ptr = NULL;
    } else {
        *p = '\0';
        ptr = ++p;
    }
    return start;
}

int main() {
```

```

char string[64];
char *token;
strcpy(string, "yashwant,chandrakant,bhosale, 1234");

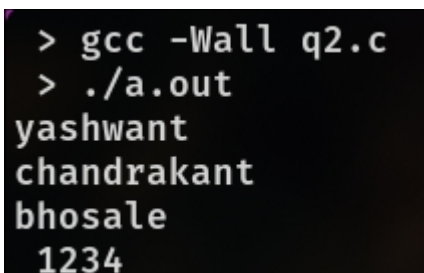
token = str_tok(string, ",");

while (token != NULL) {
    printf("%s\n", token);
    token = str_tok(NULL, ",");
}

return 0;
}

```

Output:



```

> gcc -Wall q2.c
> ./a.out
yashwant
chandrakant
bhosale
1234

```

Q3. Write a function which finds the longest possible subsequence of one string into another and returns the length + pointer to the subsequence.

```

#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int len;
    char *seq;
} result;

result *str(char *str1, char *str2) {
    char *p = str1;
    int max = 0;
    char *max_seq = NULL;
    while(*p != '\0') {
        int count = 0;
        while(p[count] == str2[count]){
            count++;
        }
        if(count > max) {
            max = count;
            max_seq = p;
        }

        count = 0;
        p++;
    }
}

```

```

    result *return_result = (result *) malloc(sizeof(result));
    return_result -> len = max;
    return_result -> seq = max_seq;
    return return_result;
}

int main() {
    char str1[] = "abcdemnopxyz";
    char str2[] = "mnotq";
    result *p = str(str1, str2);
    printf("Longest subsequence: %s,\ncharacters matched: %d\n", p-
>seq, p->len);
    return 0;
}

```

OUTPUT:

```

> gcc -Wall q3.c
> ./a.out
Longest subsequence: mnopxyz,
characters matched: 3

```

Q4. Write a function to find gcd() of 2 numbers.

```

#include <stdio.h>

int gcd(int a, int b) {
    int gcd=0;
    for(int i = 1; i<=a && i<=b; i++) {
        if(a%i == 0 && b%i == 0) {
            gcd = i;
        }
    }
    return gcd;
}

int main() {
    int a, b;
    printf("Enter two numbers: ");
    scanf("%d%d", &a, &b);
    printf("GCD of %d and %d is %d\n", a, b, gcd(a, b));
    return 0;
}

```

Output:

```

$ gcc -Wall q4.c
$ ./a.out
Enter two numbers: 12 20
GCD of 12 and 20 is 4

```

Q5. Write a function to find lcm() of 2 numbers.

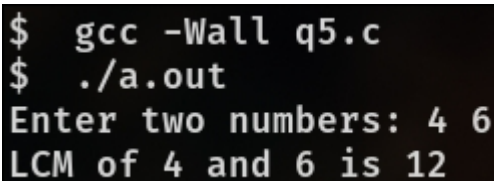
Code:

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

int lcm(int num1, int num2) {
    for(int i = 1; i <= num1*num2; i++) {
        if(i%num1 == 0 && i%num2 == 0) {
            return i;
        }
    }
    return num1*num2;
}

int main(){
    int num1, num2;
    printf("Enter two numbers: ");
    scanf("%d %d", &num1, &num2);
    printf("LCM of %d and %d is %d\n", num1, num2, lcm(num1, num2));
    return 0;
}
```

Output:



```
$ gcc -Wall q5.c
$ ./a.out
Enter two numbers: 4 6
LCM of 4 and 6 is 12
```

Q6. Write a function to convert a decimal number to a binary number and return the binary representation in a string.

Code:

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

void reverse(char *str) {
    int len = strlen(str);
    char *p = str;
    char *q = str + len - 1;

    for (int i = 0; i < len / 2; i++) {
        char c = *p;
        *p = *q;
        *q = c;
        p++;
        q--;
    }
}

char *decimal_to_binary(int n) {
```

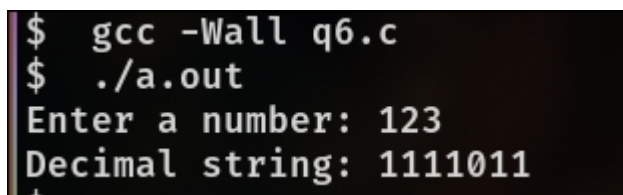
```

    char *str = (char *)malloc(sizeof(char) * 64);
    int i = 0;
    while (n > 0) {
        str[i] = n % 2 + '0';
        n = n / 2;
        i++;
    }
    str[i] = '\0';
    reverse(str);
    return str;
}

int main() {
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);
    printf("Decimal string: %s\n", decimal_to_binary(n));
    return 0;
}

```

Output:



```

$ gcc -Wall q6.c
$ ./a.out
Enter a number: 123
Decimal string: 1111011

```

Q7. Write your own code for following library functions:

strcasecmp strsep strcasecmp strcoll

code:

// Full code in 612303039/string-assingments/q7.c

```

char toLowerCase(char c) {
    if (c >= 'A' && c <= 'Z') {
        return c + ('a' - 'A');
    } else {
        return c;
    }
}

int str_case_cmp(char *s1, char *s2) {
    int i = 0;
    while (s1[i] != '\0' || s2[i] != '\0') {
        if (toLowerCase(s1[i]) == toLowerCase(s2[i])) {
            i++;
            continue;
        } else {
            return (toLowerCase(s1[i]) - toLowerCase(s2[i]));
        }
    }
    return 0;
}

int str_coll(char *s1, char *s2) {

```

```

int i = 0;
while (s1[i] != '\0' || s2[i] != '\0') {
    if (s1[i] == s2[i]) {
        i++;
        continue;
    } else {
        return s1[i] - s2[i];
    }
}
return 0;
}

char *str_sep(char **str, const char *delim) {
    char *p = *str;
    char *start = *str;
    if (p == NULL) {
        return NULL;
    }
    // Find first occurrence of delim in str
    while (*p != '\0' && *p != *delim) {
        p++;
    }

    if (*p == '\0') {
        *str = NULL;
        return start;
    } else {
        *p = '\0';
        *str = p + 1;
        return start;
    }
}

```

Output:

```

$ ./a.out
1.str_case_cmp
2.str_coll
3.str_sep
4.Exit
Enter option: 1
Enter two strings: hello Hello
str_case_cmp: 0
Strings are equal
Enter option: 2
Enter two strings: hello Hello
str_coll: 32
String 1 is greater than String 2
Enter option: 3
Enter string and delimiter: yashwant,chandrakant,bhosale ,
Token: yashwant
Token: chandrakant
Token: bhosale
Enter option: 4

```

Q9. Write following functions: sine, sine-inverse, cosine, cosine-inverse. Then using sine and cosine, write tan() function. Check whether calling sine(sine-inverse(x)) on your own functions gives you x.

code:

```
#include <stdio.h>
#include <math.h>

double sine(double x) {
    if(x > 3.14159265) {
        x = x - 3.14159265;
    }
    double sinex, term;
    int i = 3, sign = -1;
    sinex = x;
    term = x;
    while(fabs(term) > 1e-8) {
        term = (term * (x * x)) / (i * (i-1));
        sinex = sinex + (sign * term);
        sign = sign * (-1);
        i+=2;
    }
    return sinex;
}

double cosine(double x) {
    if(x > 3.14159265) {
        x = x - 3.14159265;
    }
    double cosinex, term;
    int i = 2, sign = -1;
    cosinex = 1;
    term = 1;
    while(term > 1e-8) {
        term = (term * (x * x)) / (i * (i-1));
        cosinex = cosinex + (sign * term);
        sign = sign * (-1);
        i+=2;
    }
    return cosinex;
}

double sine_inverse(double x) {
    if(fabs(x) > 1) {
        printf("Invalid Input!\n");
        return 0;
    }
    double sine_inversex, term;
    int i = 0;

    sine_inversex = x;
    term = x;

    while(fabs(term) > 1e-8) {
```

```

        term = (((2*i + 1) * (2*i + 1) * (x*x) * term)/((2*i + 2) * (2*i
+ 3)));
        sine_inversex += term;
        i++;
    }
    return sine_inversex;
}

double cosine_inverse(double x) {
    if(fabs(x) > 1) {
        printf("Invalid input!\n");
        return 0;
    }
    return (1.57079632 - sine_inverse(x));
}

int main() {
    double x, result;
    printf("Enter x: ");
    scanf("%lf", &x);
    int option;
    printf("1. Calculate Sinex\n2. Calculate Cosinex\n3. Calculate sine
inverse x.\n4. Calculate cosine inverse x.\n5. Calculate tanx.\n6.
Calculate sin^-1(sin(x)).\n");
    printf("Enter option: ");
    scanf("%d", &option);
    switch(option) {
        case 1: {
            result = sine(x);
            break;
        }
        case 2: {
            result = cosine(x);
            break;
        }
        case 3: {
            result = sine_inverse(x);
            break;
        }
        case 4: {
            result = cosine_inverse(x);
            break;
        }
        case 5: {
            result = sine(x) / cosine(x);
            break;
        }
        case 6: {
            result = sine_inverse(sine(x));
            break;
        }
        default: {
            printf("Invalid option!");
            result = -1;
            break;
        }
    }
}

```



```

    }
}
printf("result = %lf\n", result);
return 0;
}

```

Output:

```

Enter x: 3.14159265
1. Calculate Sinex
2. Calculate Cosinex
3. Calculate sine inverse x.
4. Calculate cosine inverse x.
5. Calculate tanx.
6. Calculate sin^-1(sin(x)).
Enter option: 2
result = -1.000000
$ ./a.out
Enter x: 1.57079632
1. Calculate Sinex
2. Calculate Cosinex
3. Calculate sine inverse x.
4. Calculate cosine inverse x.
5. Calculate tanx.
6. Calculate sin^-1(sin(x)).
Enter option: 1
result = 1.000000
$ ./a.out
Enter x: 0.7071067
1. Calculate Sinex
2. Calculate Cosinex
3. Calculate sine inverse x.
4. Calculate cosine inverse x.
5. Calculate tanx.
6. Calculate sin^-1(sin(x)).
Enter option: 3
result = 0.785398
$ ./a.out
Enter x: 0.7071067
1. Calculate Sinex
2. Calculate Cosinex
3. Calculate sine inverse x.
4. Calculate cosine inverse x.
5. Calculate tanx.
6. Calculate sin^-1(sin(x)).
Enter option: 5
result = 0.854510
$ ./a.out
Enter x: 0.785398
1. Calculate Sinex
2. Calculate Cosinex
3. Calculate sine inverse x.
4. Calculate cosine inverse x.
5. Calculate tanx.
6. Calculate sin^-1(sin(x)).
Enter option: 6
result = 0.785398

```

Q11. Write a program to reverse the digits of an integer and store the result as another integer

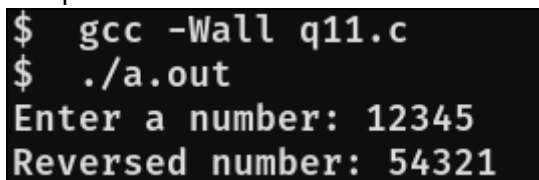
code:

```
#include <stdio.h>
#include <math.h>

int reverse_int(int n) {
    int sign_flag = 0;
    if(n<0){
        n = -n;
        sign_flag = 1;
    }
    int reverse = 0;
    while(n > 0) {
        reverse = reverse * 10 + n%10;
        n /= 10;
    }
    return sign_flag ? reverse * (-1) : reverse;
}

int main() {
    int n, reversed;
    printf("Enter a number: ");
    scanf("%d", &n);
    reversed = reverse_int(n);
    printf("Reversed number: %d\n", reversed);
    return 0;
}
```

output:



```
$ gcc -Wall q11.c
$ ./a.out
Enter a number: 12345
Reversed number: 54321
```

Q12. Write a program which reads a string and if the string has all digits in it, then derives the integer it represents.

Code:

```
#include <stdio.h>

int parse_int(char *num_string){
    int n = 0, ptr = 0, sign=1;
    if(num_string[0] == '-') {
        sign = -1;
        ptr++;
    }
    while(num_string[ptr] >= '0' && num_string[ptr] <= '9' &&
num_string[ptr] != '\0'){
        n = n * 10 + (num_string[ptr] - '0');
        ptr++;
    }
    return n * sign;
}
```

```

int main() {
    char num_str[64];
    printf("Enter a number string: ");
    scanf("%s", num_str);
    printf("%d\n", parse_int(num_str));
    return 0;
}

```

Output:

```

$ gcc -Wall q12.c
$ ./a.out
Enter a number string: 1234
1234

```

Q13. Write a function which does the following

void rev(char *str);

Reverses the string "str" in place (without using another string)

code:

```

#include <stdio.h>
#include <string.h>
void rev(char *str) {
    for(int i = 0; i < strlen(str)/2; i++){
        str[i] = str[i] + str[strlen(str) - i - 1];
        str[strlen(str) - i - 1] = str[i] - str[strlen(str) - i - 1];
        str[i] -= str[strlen(str) - i - 1];
    }
    return;
}

int main() {
    char str[128];
    printf("Enter a string : ");
    scanf("%s", str);
    rev(str);
    printf("reversed : %s\n", str);
    return 0;
}

```

output:

```

$ gcc -Wall q13.c
$ ./a.out
Enter a string : Yashwant
reversed : tnawhsaY

```

Q14. Write a function which cuts a string given by "str" on the character given in "ch" and returns the first such word.

Code:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
char *cutonchar(char *str, char ch) {
    char str_buffer[64];
    int i = 0;
    while(str[i] != ch && str[i] != '\0') {
        str_buffer[i] = str[i];
        i++;
    }
    if(str[i] == ch) {
        str_buffer[i++] = ch;
        str_buffer[i] = '\0';
        char *return_str = (char *) malloc(sizeof(char) * i);
        strcpy(return_str, str_buffer);
        return return_str;
    }
    return NULL;
}

int main() {
    char str[] = "something";
    printf("%s\n", cutonchar(str, 'e'));
    return 0;
}
```

Output:

```
$ gcc -Wall q14.c
$ ./a.out
some
```