Repo Link: [C-Course-IEEE/Session2/Task at main · AhmEeAli2003/C-Course-IEEE (github.com)](https://github.com/AhmEeAli2003/C-Course-IEEE/tree/main/Session2/Task)

Task1:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

int main(void)

{

signed int number = INT\_ZERO;

printf("Input number: ");

scanf("%d", &number);

/\* Solution using simple if \*/

if(INT\_ZERO == number)

{

printf("%d is equal to zero\n", number);

}

if(number > INT\_ZERO)

{

printf("%d is positive\n", number);

}

if(number < INT\_ZERO)

{

printf("%d is negative\n", number);

}

/\* Solution using if else \*/

/\*

if(INT\_ZERO == number)

{

printf("%d is equal to zero\n", number);

}

else if(number > INT\_ZERO)

{

printf("%d is positive\n", number);

}

else

{

printf("%d is negative\n", number);

}

\*/

return SUCCESS\_RET;

}

Task2:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

#define A\_CAPITAL 'A'

#define Z\_CAPITAL 'Z'

#define A\_SMALL 'a'

#define Z\_SMALL 'z'

#define CHAR\_ZERO '0'

#define CHAR\_NINE '9'

int main(void)

{

unsigned char character = INT\_ZERO;

printf("Input any character: ");

scanf("%c", &character);

if((character >= A\_CAPITAL && character <= Z\_CAPITAL) || (character >= A\_SMALL && character <= Z\_SMALL))

{

printf("%c is alphabet\n", character);

}

else if(character >= CHAR\_ZERO && character <= CHAR\_NINE)

{

printf("%c is digit\n", character);

}

else

{

printf("%c is special character\n", character);

}

return SUCCESS\_RET;

}

Task3:

#include <stdio.h>

#include <math.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

#define TEN 10

int main(void)

{

unsigned int number, swappedNumber, digitsNumber, lastDigit, firstDigit;

number = swappedNumber = digitsNumber = lastDigit = firstDigit = INT\_ZERO;

printf("Input any number: ");

scanf("%d", &number);

/\* Suppose that user input 12345 \*/

lastDigit = number % TEN; /\* lastDigit = 5 \*/

digitsNumber = (int) log10(number); /\* digitsNumber = 4 \*/

firstDigit = (int) (number / pow(TEN, digitsNumber)); /\* last digit = 12345 / 10000 = 1 \*/

swappedNumber = lastDigit; /\* swappedNumber = 5 \*/

swappedNumber \*= (int) round(pow(TEN, digitsNumber)); /\* swappedNumber = 5 \* 10000 = 50000 \*/

swappedNumber += number % (int) round(pow(TEN, digitsNumber)); /\* swappedNumber = 50000 + (12345 % 10000) = 52345 \*/

swappedNumber -= lastDigit; /\* swappedNumber = 52345 - 5 = 52340 \*/

swappedNumber += firstDigit; /\* swappedNumber = 52340 + 1 = 52341 \*/

printf("Number after swapping first and last digit = %d\n", swappedNumber);

return SUCCESS\_RET;

}

Task4:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

#define ONE 1

#define TWO 2

int main(void)

{

signed int number = INT\_ZERO;

signed int first\_iterator = INT\_ZERO, second\_iterator = INT\_ZERO;

unsigned short isPrime = INT\_ZERO;

printf("Input any number: ");

scanf("%d", &number);

printf("Prime factors of %d: ", number);

/\* Find all prime factors \*/

for(first\_iterator = TWO; first\_iterator <= number / TWO; first\_iterator++)

{

/\* check if 'first\_iterator' is factor of number \*/

if(INT\_ZERO == number % first\_iterator)

{

isPrime = ONE;

/\* Check if this factor is prime or not \*/

for(second\_iterator = TWO; second\_iterator <= first\_iterator / TWO; second\_iterator++)

{

if(INT\_ZERO == first\_iterator % second\_iterator)

{

isPrime = INT\_ZERO;

break;

}

}

if(isPrime)

{

printf("%d, ", first\_iterator);

}

}

}

printf("\b\b.");

return SUCCESS\_RET;

}

Task5:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

#define ONE 1

#define TWO 2

int main(void)

{

unsigned int number = INT\_ZERO, sum\_primes = INT\_ZERO;

unsigned short iterator = INT\_ZERO;

printf("Input upper limit: ");

scanf("%d", &number);

if(ONE == number)

{

sum\_primes = ONE;

}

else

{

for(iterator = TWO; iterator <= number; iterator++)

{

if(INT\_ZERO == number % iterator)

{

sum\_primes += iterator;

}

}

}

printf("Sum of prime numbers between 1-%d: %d", number, sum\_primes);

return SUCCESS\_RET;

}

Task6:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

int main(void)

{

unsigned char day = INT\_ZERO;

printf("Input day number(1-7): ");

scanf("%c", &day);

switch(day)

{

case '1':

printf("Saturday\n");

break;

case '2':

printf("Sunday\n");

break;

case '3':

printf("Monday\n");

break;

case '4':

printf("Tuseday\n");

break;

case '5':

printf("Wednesday\n");

break;

case '6':

printf("Thuesday\n");

break;

case '7':

printf("Friday\n");

break;

default:

printf("Invalid input\n");

}

return SUCCESS\_RET;

}

Task7:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

#define FALSE 0

#define TRUE 1

int main(void)

{

signed int first\_number = INT\_ZERO, second\_number = INT\_ZERO;

unsigned short isMaximum = INT\_ZERO;

printf("Input first number: ");

scanf("%d", &first\_number);

printf("Input second number: ");

scanf("%d", &second\_number);

isMaximum = first\_number > second\_number;

switch(isMaximum)

{

case TRUE:

printf("Maximum: %d\n", first\_number);

break;

case FALSE:

printf("Maximum: %d\n", second\_number);

break;

}

return SUCCESS\_RET;

}

Task8:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

#define TWO 2

int main(void)

{

unsigned int number = INT\_ZERO;

printf("Input number: ");

scanf("%d", &number);

(number % TWO)? printf("%d is odd\n", number):printf("%d is even\n", number);

return SUCCESS\_RET;

}

Task9 = Task8.

Task10:

#include <stdio.h>

#define SUCCESS\_RET 0

#define INT\_ZERO 0

#define FOUR 4

#define ONE\_HUNDRED 100

#define FOUR\_HUNDRED 400

int main(void)

{

unsigned short year = INT\_ZERO;

printf("Input year: ");

scanf("%hu", &year);

/\*

If a year is exactly divisible by 4 and not divisible by 100 then its Leap year.

Else if year is exactly divisible 400 then its leap year. Else its a Common year.

\*/

printf("%hu is %s\n", year, ((INT\_ZERO == year % FOUR) && (INT\_ZERO != year % ONE\_HUNDRED))? "Leap Year":

(INT\_ZERO == year % FOUR\_HUNDRED)? "Leap Year":"Common Year");

return SUCCESS\_RET;

}