CSE 220
C Programming

Examples of loop structure
Announcement

• The solution to midterm exam is on course website
Example (I)

• Write a program that reads an integer $n$ from the user input and compute a factorial of $n$ ($n!$).
Example (I)

• Write a program that reads an integer \( n \) from the user input and compute the factorial of \( n \) \((n!)\).

    Factorial of number is defined as:

\[
    n! = 1 \times 2 \times \ldots \times n
\]

\[
4! = 1 \times 2 \times 3 \times 4 = 24
\]

• For example:
  – input: 4
  – output: (Factorial value) : 24

\[
5! = 1 \times 2 \times 3 \times 4 \times 5 = 120
\]

• For example:
  – input: 5
  – output: (Factorial value) : 120

Factorial of zero = 1
• Write a program that reads an integer n from the user input and compute a factorial of n (n!).

Steps:

1) Declare two integers num and fact.

2) Prompt the message to enter any number to calculate the factorial.

3) Allow the user to enter number using scanf().

4) Calculate the factorial value and return that.

5) Print the returned value to the screen using printf() command.
Factorial of a number: for loop

```c
#include <stdio.h>

int main()
{
    int fact, num, i;
    fact = 1;
    printf("Enter an integer number: \n");
    scanf("%d", &num);

    for(i = 1; i <= num; i++)
    {
        fact = fact * i;
    }

    printf("Factorial of %d is: %d\n", num, fact);

    return 0;
}
```
Factorial of a number: while loop

```c
#include <stdio.h>

int main()
{
    int fact, num, i;
    fact = 1;
    i = 1;
    printf("Enter an integer number: \n");
    scanf("%d", &num);

    while( i <= num )
    {
        fact = fact * i;
        i = i + 1;
    }

    printf("Factorial of %d is: %d\n", num, fact);

    return 0;
}
```
Declare and Initialize control variables

```c
#include <stdio.h>

int main()
{
    int fact, num, i;
    fact = 1;
    printf("Enter an integer number: \n");
    scanf("%d", &num);

    for(i = 1; i <= num; i++)
        fact = fact * i;

    printf("Factorial of %d is: %d\n", num, fact);

    return 0;
}
```

Visible in all program:

Visible only in for loop:

(I)

(II)
#include <stdio.h>

int main()
{
    int fact, num;
    fact = 1;
    printf("Enter a number");
    scanf("%d", &num);
    for (int i = 1; i <= num; i++)
    {
        fact = fact * i;
    }
    printf("Final value of i is: %d\n", i);
    printf("Factorial of %d is: %d\n", num, fact);
    return 0;
}
Declare and Initialize control variables

```c
#include <stdio.h>
int main()
{
    int fact, num, i;
    fact = 1;
    printf("Enter a number");
    scanf("%d",&num);
    for (i = 1 ; i <= num; i++)
    {
        fact = fact * i;
    }
    printf("Final value of i is: %d\n",i);
    printf("Factorial of %d is: %d\n",num,fact);
    return 0;
}
```
Declare and Initialize

- Can declare a variable inside expr1
- The variable is visible only inside loop

```c
for (int i=0;  i<3; i++)
{
    printf("i is %d\n", i);
}
printf("Now i is: %d\n", i);
```

Wrong!

```c
fact.c:10:37: error: use of undeclared identifier 'i'
    printf("Final value of i is: %d\n",i);
```
Declare and Initialize

```c
for (int i=0; i<3; i++)
{
    printf("i is %d\n", i);
}
printf("Now i is: %d\n", i);
```

Wrong!

```c
int i;
for (i=0; i<3; i++)
{
    printf("i is %d\n", i);
}
printf("Now i is: %d\n", i);
```

Right.
Question?

```
#include <stdio.h>

int main()
{
    int i = 10, sum = 0;
    for(int i = 1; i <= 5; i++)
    {
        sum = sum + i;
    }
    printf("Final value of i is: %d\n", i);
    printf("Final value of sum is: %d\n", sum);
    return 0;
}
```
Question?

```
#include <stdio.h>

int main()
{
    int i = 10, sum = 0;
    for(int i = 1; i <= 5; i++)
    {
        sum = sum + i;
    }
    printf("Final value of i is: %d\n", i);
    printf("Final value of sum is: %d\n", sum);
    return 0;
}
```

Final value of i is: 10
Final value of sum is: 15
Factorial of a number

The factorial of a number, denoted as $n!$, is the product of all positive integers less than or equal to $n$. For example, $5! = 5 	imes 4 	imes 3 	imes 2 	imes 1 = 120$.

When calculating $100!$, the result is a very large number. In fact, $100!$ is so large that it does not fit in a standard integer variable in most programming languages.

This is because the factorial grows very rapidly. The exact factorial of a number can be computed using mathematical formulas, but for large numbers, computational limitations come into play.

To justify why this happens, consider the following:

- **Mathematical Insight**: The factorial function grows faster than any polynomial function. This means that even for moderately large values of $n$, the factorial becomes astronomically large.
- **Computational Limitations**: Most computer systems have limitations on the size of integers that can be stored. Integers are typically limited to a fixed number of bits, which restricts the maximum value they can store.

Thus, when calculating $100!$, the result is too large to be stored in a standard integer variable, leading to overflow errors or the display of a large, albeit truncated, number.
Exercise: Factorial of a number

#include <stdio.h>

int main()
{
    int fact; // What data type should it be?

    int num, i;
    fact = 1;
    printf("Enter a number: ");
    scanf("%d", &num);
    for (i = 1; i <= num; i++)
    {
        fact = fact * i;
    }

    printf("%d! = \n", num, fact); // What format specifier is needed? (%d? %c? ... ?)
return 0;
}

Fill out the two blank spaces in order to have this factorial code working for input of ‘num = 20’. The results should be ‘2,432,902,008,176,640,000’.

(You just need to specify the data type and the format specifier)
Example (II)

• Write a program that reads an integer $n$ from the user input and prints all the prime numbers less than input.

• For example:
  – input:  7
  – output:  5,3,2

• For example:
  – input:  8
  – output:  7,5,3,2
Example (II)

• Write a program that reads an integer \( n \) from the user input and prints all the prime numbers less than input.

• For example:
  – input: 7
  – output: 5, 3, 2
Read $n$ from user

$n --$

Is $n$ prime?

Yes

print $n$

No
Example (II)

A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself (i.e., it has exactly two positive divisors, 1 and the number itself).

- 5 divides 1 and 5
- 4 divides 1, 2, and 4
- 13 divides 1 and 13

Read \( n \) from user

\( n \) --

Is \( n \) prime?

Yes

print \( n \)

No
Solution (I)

```
Read n from user

n --

Is n prime?

Yes

print n

No

j = num
counter = 0

r = n%j

r == 0

counter++

Yes

j >= 1

No

j--

j >= 1

r == 0

Yes

counter == 2

Yes

n is prime

No
```

Is n prime?

Yes

print n

No

j --

r == 0

Yes

counter++

Yes

n is prime
#include<stdio.h>
int main()
{
    int i;
    int num;
    int count = 0;
    int j;
    printf("Please enter a number: ");
    scanf("%d", &num);
Solution (I)

```c
#include<stdio.h>

int main()
{
    int i;
    int num;
    int count = 0;
    int j;
    printf("Please enter a number: ");
    scanf("%d", &num);

    for (j = num; j >= 1; j--)
    {
    }
```
```
#include<stdio.h>
int main()
{
    int i;
    int num;
    int count = 0;
    int j;
    printf("Please enter a number: ");
    scanf("%d",&num);

    for (j = num; j >= 1; j--)
    {
        for (i = 1; i <= j; i++)
        {

        }
    }
}
```
```c
#include<stdio.h>
int main()
{
    int i;
    int num;
    int count = 0;
    int j;
    printf("Please enter a number: ");
    scanf("%d",&num);

    for (j = num; j >= 1; j--)
    {
        for (i = 1; i <= j; i++)
        {
            if (j%i == 0)
            {
                count = count + 1;
            }
        }

        if (count == 2)
        {
            printf(" %d is the prime number\n", j);
        }
    }
    count = 0;
}
```
```c
#include<stdio.h>
int main()
{
    int i;
    int num;
    int count = 0;
    int j;
    printf("Please enter a number: ");
    scanf("%d",&num);

    for (j = num; j >= 1; j--)
    {
        for (i = 1; i <= j; i++)
        {
            if (j%i == 0)
            {
                count = count + 1;
            }
        }

        if (count == 2)
        {
            printf(" %d is the prime number\n", j);
        }
    }
    count = 0;
}
```
Solution (II)

Read n from user

Is n prime?

Yes

print n

No

n -

n is not prime

i = 2

i < n

r = n%i

r == 0

Yes

n is prime

No

i++

Yes

n is not prime

r == 0

No

Yes
#include <stdio.h>
int main()
{
    int n = 0, num = 0;
    int flag;
    printf("Enter the number: \n");
    scanf("%d", &n);

    return 0;
}
Solution (II)

```c
#include <stdio.h>
int main()
{
    int n = 0, num = 0;
    int flag;
    printf("Enter the number: \n");
    scanf("%d", &n);

    for (j = num; j >= 1; j--)
    {
    }

    return 0;
}
```
```c
#include <stdio.h>
int main()
{
    int n = 0, num = 0;
    int flag;
    printf("Enter the number: \n");
    scanf("%d", &n);

    for (j = num; j >= 1; j--)
    {
        /* check if the num is prime or not */
        flag = 1;
        for (int i = 2; i <= j - 1; i++)
        {
            /* code */
        }
    }

    return 0;
}
```
```c
#include <stdio.h>
int main()
{
    int n = 0, num = 0;
    int flag;
    printf("Enter the number: \
");
    scanf("%d",&n);

    for ( j = num; j>= 1; j-- )
    {
        /* check if the num is prime or not */
        flag = 1;
        for ( int i = 2 ; i <= j - 1 ; i++ )
        {
            if ( j % i == 0 )
            {
                flag = 0;
            }
            if(flag)
            {
                printf("The number %d is prime\n",j);
            }
        }
    }
    return 0;
}
```
Solution (II)

#include <stdio.h>

int main()
{
    int n = 0, num = 0;
    int flag;
    printf("Enter the number: \n");
    scanf("%d", &n);

    for (j = num; j >= 1; j--)
    {
        /* check if the num is prime or not */
        flag = 1;
        for (int i = 2; i <= j - 1; i++)
        {
            if (j % i == 0)
            {
                flag = 0;
            }
        }
        if (flag)
        {
            printf("The number %d is prime\n", j);
        }
    }
    return 0;
}
Exiting from a loop