ព្រះរាជាណាចក្រកម្ពុជា

ជាតិ សាសនា ព្រះមហាក្សត្រ

វិទ្យាស្ថានបច្ចេកវិទ្យាកម្ពុជា

ដេប៉ាតឺម៉ង់ថ្នាក់ឆ្នាំសិក្សាមូលដ្ឋាន

TP6-Using loop while

Subject: Data strucure & Programming

Student name	ID	score
1.SOUNG VICHKA	e20240672	…..

Teacher: Mr.ROEUN Pachroth

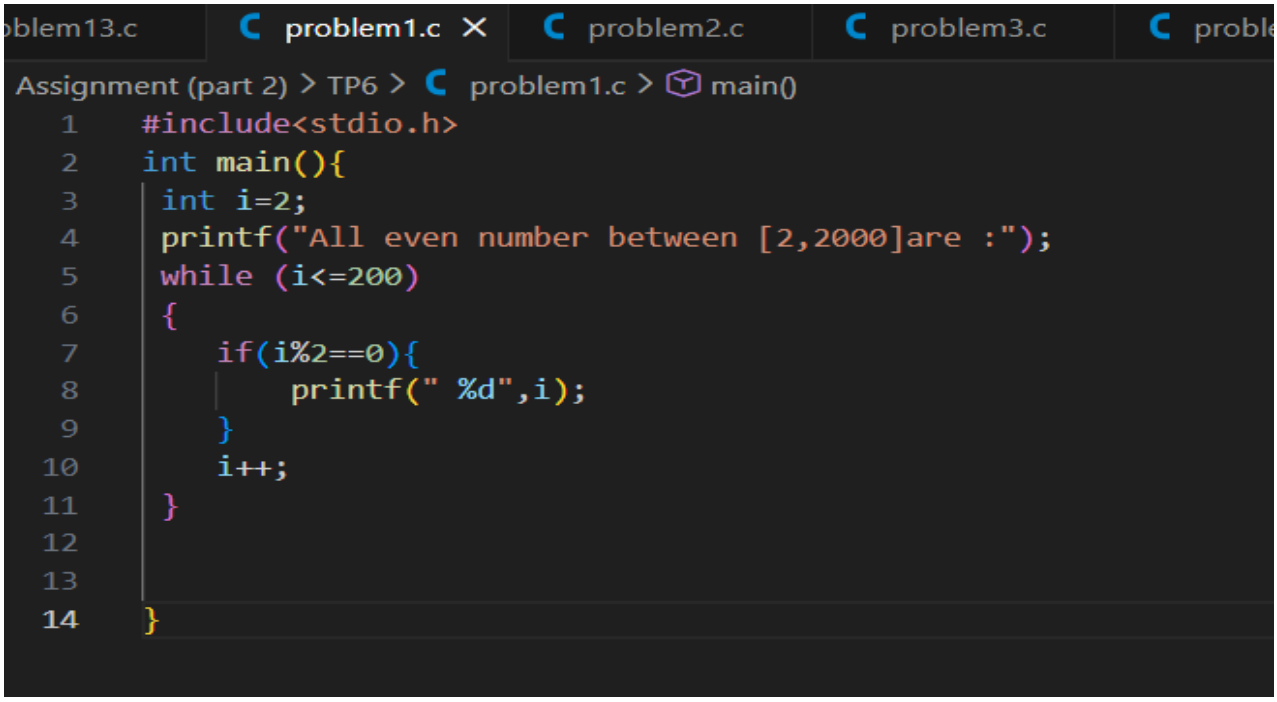
Problem1: Write a program to display all even numbers in this range [2, 2000] using a while loop.

Example:

Output:

All even numbers between [2, 2000] are: 2 4 6 8 10 …… 2000

Code:



Explain:

- **int i = 2;**

Starts the variable i at 2 (the first even number).

- **printf("All even number between [2,2000]are :");**
  Prints a message to the screen. Notice it says [2,2000], but the loop condition is i <= 200. So the program actually prints **even numbers from 2 to 200**, not 2000.
- **while (i <= 200)**
  A loop that keeps running as long as i is less than or equal to 200.
- **if (i % 2 == 0)**
  Checks if i is divisible by 2. If true, i is even.
- **printf(" %d", i);**
  Prints the current even number.
- **i++;**
  Increases i by 1 each time the loop runs.

Result:

```
PS D:\SDP week5>  & 'c:\Users\User\.vscode\extensions\ms-vscode.cpptools-1.29.2-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=
Microsoft-MIEngine-In-3vf5snsv.f0c' '--stdout=Microsoft-MIEngine-Out-tnzdmscv.v3b' '--stderr=Microsoft-MIEngine-Error-2rnifz4z.2ue' '--pid=Micr
osoft-MIEngine-Pid-vbqwbuyb.djy' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
All even number between [2,2000]are : 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 7
4 76 78 80 82 84 86 88 90 92 94 96 98 100 102 104 106 108 110 112 114 116 118 120 122 124 126 128 130 132 134 136 138 140 142 144 146 148 150 1
52 154 156 158 160 162 164 166 168 170 172 174 176 178 180 182 184 186 188 190 192 194 196 198 200
PS D:\SDP week5>
```

Problem2: Write a program that prints the powers of 2 up to 2^10 (1024) using a while loop.

Example:

Output:

2^0 = 0
2^1 = 2
2^2 = 4
2^3 = 8
2^4 = 16
2^5 = 32
2^6 = 64
2^7 = 128
2^8 = 256
2^9 = 512
2^10 = 1024

Code:

5.c Assignment (part 2) | problem9.c | problem10.c | problem11.c

Assignment (part 2) > TP6 > problem2.c > main()

```c
#include<stdio.h>
#include<math.h>
int main(){
    int i=0;
    while (i<=10)
    {
        int sum=pow(2,i);
        printf("2^%d=%d\n",i,sum);
        i++;
    }

}
```

- **#include<stdio.h>**
  Lets you use printf for output.
- **#include<math.h>**
  Provides math functions like pow() (power function).
- **int i = 0;**
  Starts the counter at 0.
- **while (i <= 10)**
  Loop continues as long as i is less than or equal to 10.
  So it runs for i = 0, 1, 2, …, 10.
- **pow(2, i)**
  Computes (2^i).
  Example:
    - pow(2,0) ⟶ 1
    - pow(2,1) ⟶ 2
    - pow(2,2) ⟶ 4
    - … up to pow(2,10) ⟶ 1024
- **printf("2^%d=%d\n", i, sum);**
  Prints the exponent and the result in the format:

    - 2^0=1
    - 2^1=2
    - 2^2=4
    - ...
    - 2^10=1024
    - **i++;**
      Increases i by 1 each loop iteration.

Result:

```
PS D:\SDP week5>  & 'c:\Users\User\.vscode\extensions\ms-vscode.cpptools-1.29.2-win32-x64\debugAdapte
Microsoft-MIEngine-In-kke1jk1n.rkh' '--stdout=Microsoft-MIEngine-Out-zj0sb1b0.yat' '--stderr=Microsoft
osoft-MIEngine-Pid-ogcaa5j5.fur' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
2^0=1
2^1=2
2^2=4
2^3=8
2^4=16
2^5=32
2^6=64
2^7=128
2^8=256
2^9=512
2^10=1024
PS D:\SDP week5>
```

Problem3: Write a program to ask a user for two integer numbers **m** and **n**. The program computes summation of numbers starting from m to n using a while loop.

Examples:

INPUT:

Enter a number m: 5
Enter a number n: 10

Output:

The sum is: 5+6+7+8+9+10 = 45

.....

INPUT:

Enter a number m: 8
Enter a number n: 2

Output:

Oop! We can not compute the summation. n must be bigger than m.

Code:

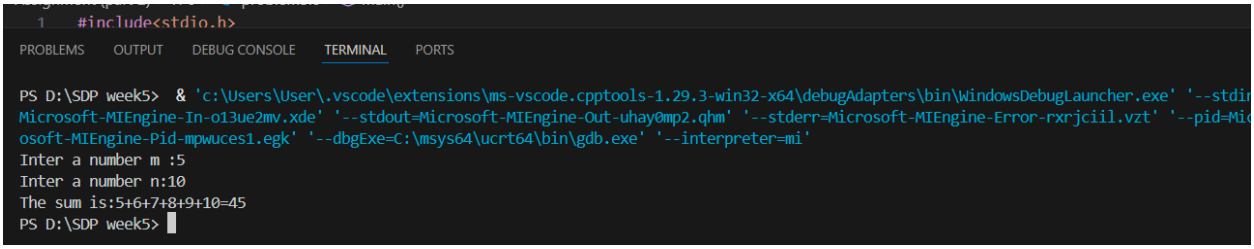
Assignment (part 2) > TP6 > problem3.c > main()

```c
#include<stdio.h>
int main(){
 int n,m,sum=0;
 printf("Inter a number m :");
 scanf("%d",&m);
 printf("Inter a number n:");
scanf("%d",&n);
if(n>=m){
    int i=m;
    printf("The sum is:");
    while (i<=n){
        sum+=i;
        printf("%d",i);
        if(i<=n){
            printf("+");
        }
        i++;
    }
    printf("=%d\n",sum);
} else{
    printf("Oop! We can not compute the summation. n must be bigger than m.");
}

}
```

1. **Input values**
   - The program asks the user to enter two integers: `m` and `n`.
2. **Check condition (`if (n >= m)`)**
   - If `n` is greater than or equal to `m`, the program will compute the sum of all integers from `m` to `n`.
   - If `n < m`, it prints an error message:
   - `Oop! We can not compute the summation. n must be bigger than m.`
3. **Summation loop (`while (i < n)`)**
   - Starts from `i = m` and keeps adding numbers until `i = n`.
   - Each number is printed, followed by a `"+"`.
4. **Final result**
   - After the loop ends, the total sum is printed.

Result:



Problem4: Write a program to ask a user to input a number. Keep asking the user to input another number if yet not input the number 0. When the user input 0, the program stops and display total summation of all input numbers.

Examples:

INPUT:

Enter a number: 5
Enter a number: 10
Enter a number: 2
Enter a number: 1
Enter a number: -1
Enter a number: 0

Output:

The sum of all input numbers is 17

.....

INPUT:

Enter a number: 20
Enter a number: 0

Output:

The sum of all input numbers is 20.

Code:

problem10.c problem11.c problem12.c problem13.c problem1.c

Assignment (part 2) > TP6 > problem4.c > ...

```c
#include<stdio.h>
int main(){
    int n;
    int sum=0;

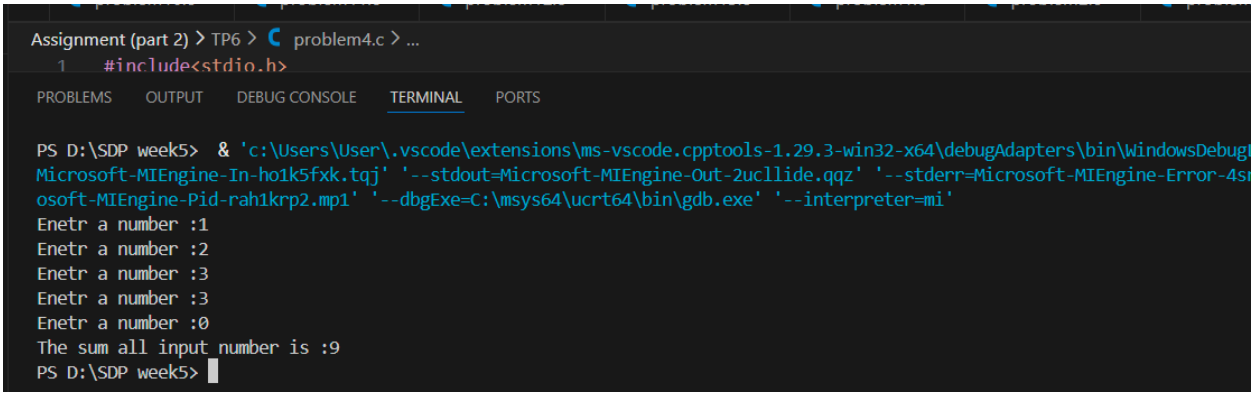
    while(1){
        printf("Enetr a number :");
        scanf("%d",&n);
        if(n==0){
            break;
        }
        sum+=n;
    }
    printf("The sum all input number is :%d ",sum);

}
```

1. **Variables**
    - n → stores the number entered by the user.
    - sum → accumulates the total of all entered numbers.
2. **while(1)**
    - This creates an **infinite loop** (runs forever) until a break statement stops it.
3. **User Input (scanf)**
    - Each time through the loop, the program asks the user to enter a number.
4. **Exit Condition (if(n == 0))**
    - If the user enters 0, the program executes break; which **stops the loop**.
5. **Summation (sum += n)**
    - If the input is not zero, the number is added to the running total.
6. **Final Output**

- After the loop ends, the program prints the total sum of all numbers entered before 0.

Result



Problem5: Write a program to ask a user for an integer number n less than 10000. The program tells the user how many digits that this number has. **Hint**: Use a while loop. Divide the number by 10 until reaching 0. Each time, try to increase a counter variable which represents the number of digits.

Examples:

INPUT:

Enter a number n: 552

Output:

This number 552 has 3 digits.

.....

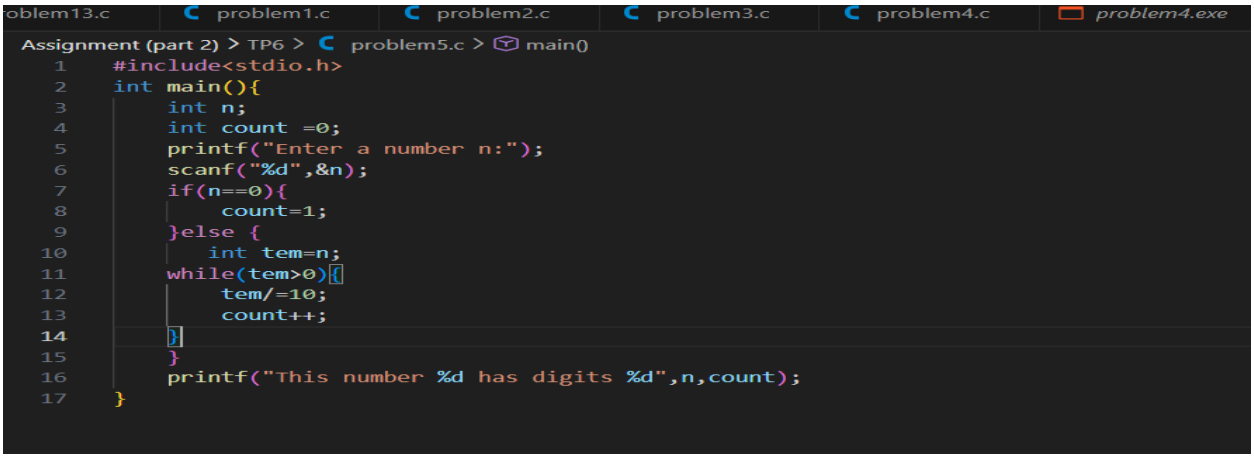
INPUT:

Enter a number n: 20021

Output:

This number 20021 has 5 digits.

Code:



1. **Input**
   - The program asks the user to enter an integer `n`.
2. **Special Case (`if(n == 0)`)**
   - If the user enters `0`, the program sets `count = 1`.
   - This is because `0` is a single-digit number.
3. **Digit Counting (`while(tem > 0)`)**
   - For any positive number, the program copies it into `tem`.
   - Each time through the loop:
     - `tem /= 10;` removes the last digit.
       Example: 123 → 12 → 1 → 0
     - `count++` increases the digit counter.
   - When `tem` becomes 0, the loop stops.
   - The number of times we divided by 10 equals the number of digits.
4. **Output**

- Finally, the program prints both the number and how many digits it has.

Result:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS D:\SDP week5>  & 'c:\Users\User\.vscode\extensions\ms-vscode.cpptools-1.29.3-win32-x64\debugAdapters\bin\Window
Microsoft-MIEngine-In-krn2bldi.yrn' '--stdout=Microsoft-MIEngine-Out-jh4gw4wz.iaz' '--stderr=Microsoft-MIEngine-Er
osoft-MIEngine-Pid-32hitns5.iid' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
Enter a number n:552\
This number 552 has digits 3
PS D:\SDP week5>
```

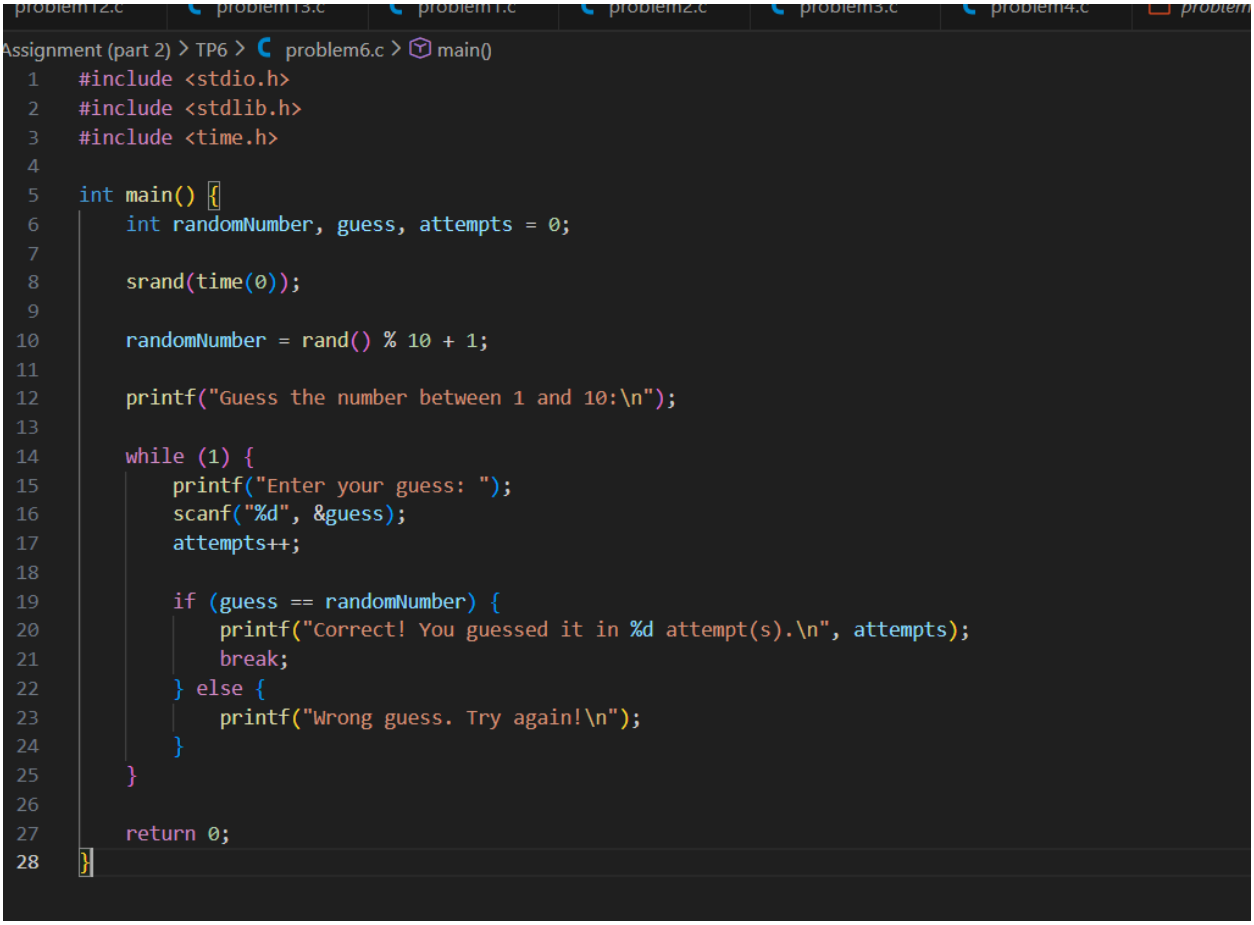
Problem6: Generate one random number. Then create a simple number guessing game where the user has to guess a number between 1 and 10. Use a while loop to continue prompting the user until they guess correctly. The program stops when the user guesses it correctly and tells the user how many that he/she has tried guessing.

Hint:
-The program randoms a number between 1 and 15. The user input a number to guess. In your program, you add a logic that when the user input is the same as the randomized number, the program stops.
-In order to generate a random number in C programming, we use the **rand()** function. To get the different number when random again, the **srand(time(0))** is used. You can find a simple code for generating a random number in the range of [1, 20].

```c
#include<stdio.h>
#include<time.h>
main(){
    srand(time(0));
    int start=1, end=20;
    int n;

    n = rand()%end + start;
    printf("%d ",n);
}
```

Code:



1. **Random Number Generation**
   - `srand(time(0));` → seeds the random number generator with the current time, so you get a different random number each run.
   - `rand() % 10 + 1;` → generates a number between 1 and 10.
   - `% 10` gives a remainder between 0–9.
   - `+ 1` shifts it to 1–10.
2. **Infinite Loop (`while(1)`)**
   - Keeps asking the user for guesses until they get it right.
   - The loop only stops when `break;` is executed.
3. **User Input**
   - `scanf("%d", &guess);` → reads the user's guess.
4. **Attempts Counter**
   - `attempts++;` → increases the number of tries each time the user guesses.
5. **Correct Guess**
   - If the guess matches `randomNumber`, the program prints success and exits the loop.

Wrong Guess

- If not equal, it prints `"Wrong guess. Try again!"` and continues looping.

Result:

