

Summary of Lesson 2

printf()

What's printf()?

- Printf is a **library function** to display messages to the **standard output(console)**
- The first parameter must be a **string(array of characters)**
- A **string** must be always surrounded by ""(Double quotation)
- You can put **%d %c %s** etc.. in the **first parameter(string)**, and **specify them with following parameters**
- You can use several **escape sequence characters** like \n(new line), \t(tab) and so on
- You can specify **accuracy of floating point number output** with .(number)

Constant and Variable

What's Constant?

- Constant is fixed value like 57, 3.14, 'o', "Volker",etc...

What's Variable?

- Variable is non-fixed value like x, i, k. name etc...

Declaration of variables

- **(data class)(signed/unsigned)(data type) name**
- You can not use **keyword**(e.g. int, while, else, if) for the name
- Variable names must **not** start with **capital characters**

Initilization of variables

- just after declaration

```
int myAge;
myAge = 26;
```
- You can initialize it at the same time

```
int myAge = 26;
```

Data type of variables

- Data type defines the content of variable(integer, floating point, or character) and its capacity/accuracy

Integer type/Mac

- int... 4 byte (+2,147,483,647 ~ -2,147,483,648)
- long... 4 byte (+2,147,483,647 ~ -2,147,483,648)
- short... 2 byte (+32767 ~ 32768)

Character type/Mac

- char... 1 byte (-128 ~ 128)

with floating point/Mac

- float... 4 byte ($\pm 2.9387e-39$ - $1.7014e+38$)
- double... 8byte

unsigned / signed

- you can put unsigned / signed before **int, short, long, char** to shift the capacity.
- you can store **bigger** value with **unsigned** but you can not use values with minus

lesson2_1.c

```
#include <stdio.h>
/* variable and its initialization*/
int main(void)
{
    int age; // declaration of variable
    age = 26; // initialization of variable
    printf("Hello Computer, I am Chikashi\nI am %d years old.", age);
    return 0;
}
```

lesson2_2.c

```
#include <stdio.h>
/* floating point variable and initialize them in the same line*/
int main(void)
{
    int age = 26;
    float money = 2.3;
    printf("I am %d years old.\nI have %f franken in my pocket",age, money);
    return 0;
}
```

lesson2_3.c

```
#include <stdio.h>
/* specify how precise the output is*/
int main(void)
{
    int age = 26;
    float money = 2.3;
    printf("I am %d years old.\nI have %.2f franken in my pocket",age, money);
    return 0;
}
```

lesson2_4.c

```
#include <stdio.h>
/* you can not change the value*/
int main(void)
{
    const int age = 26;
    float money = 2.3;
    age = 20;
    printf("I am %d years old.\nI have %.2f franken in my pocket",age, money);
    return 0;
}
```

lesson2_5.c

```
#include <stdio.h>
/* declaration and initialization of two values at the same time*/
int main(void)
{
    int age = 26,yearOfBirth = 1979;
    float money = 2.3;
    printf("I am %d years old.\nand my year of birth is %d.\nI have %.2f franken in my pocket",age, yearOfBirth, money);
    return 0;
}
```

lesson2_6.c

```
#include <stdio.h>

/* initialization at the same time*/
int main(void)
{
    int age = 26, yearOfBirth = 1979;
    float money = 2.3;
    printf("I am %d years old.\nand my year of birth is %d.\nI have %.2f franken in my pocket",age, yearOfBirth, money);
    return 0;
}
```

lesson2_7.c

```
#include <stdio.h>
/* The first C calculation in your life*/
int main(void)
{
    int age = 26, yearOfBirth = 1979, thisYear;
    thisYear = yearOfBirth + age;
    printf("I am %d years old.\nand my year of birth is %d.\nSo this year might be A.D. %d", age, yearOfBirth, thisYear);
    return 0;
}
```

lesson2_8.c

```
#include <stdio.h>
/* experiment of data type short*/
int main(void)
{
    short number;
    number = 50000;
    printf("%d", number);
    return 0;
}
```

lesson2_9c

```
#include <stdio.h>
/* experiment of data type unsigned short*/
int main(void)
{
    unsigned short number;
    number = 50000;
    printf("%d", number);
    return 0;
}
```

lesson2_10.c

```
#include <stdio.h>
/* you can store numbers also in char, but must be very small number*/
int main(void)
{
    char number;
    number = 120;
    printf("%d", number);
    return 0;
}
```

lesson2_11.c

```
#include <stdio.h>
/*vice versa, for computer characters are just a number*/
int main(void)
{
    int number;
    number = 120;
    printf("%c",number);
    return 0;
}
```

lesson2_12.c

```
#include <stdio.h>
/* So precise, I can not remember....*/
int main(void)
{
    float pi = 3.14159265358979;
    printf("%.14f", pi);
    return 0;
}
```

lesson2_13.c

```
#include <stdio.h>
/* But I can..*/
int main(void)
{
    double pi = 3.14159265358979;
    printf("%.14f", pi);
    return 0;
}
```

lesson2_14.c

```
#include <stdio.h>
/* cast it*/
int main(void)
{
    double pi =3.14159265358979;
    int api;
    api = (int)pi;
    printf("PI is about %d",api);
    return 0;
}
```