Loop (2)

Department of Information System
SoICT, HUST

`do...while` statement

```
do {
   statements;
} while ( <condition> );
```

- is used to repeat a statement or a block of code several times

- The loop will always be executed at least once, before the test is made to determine whether it should continue

Example

- Calculate the sum of an integer array (ver 3)

Algorithm: (version 3)

```
sum = 0
do
{
    input aNum
    add aNum to sum
} while (aNum!=0)
output sum
```

```
#include <stdio.h>
int main()
{
  int aNum, sum = 0;
  do
{
  scanf("%d", &aNum) ;
    sum += aNum;
  } while (aNum!=0);
  printf("Sum is %d\n", sum);
  return 0;
}
```

Using do…while to verify input data

```
int n;

do {
  printf("Input a positive number: ");
  //n = -1;
  fflush(stdin);
  if (scanf("%d", &n) <0 )
     printf("Input data is not a positive
  number\n");
}while (n>0);

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

Result

- Input a positive number: -2
- Input a positive number: 5
- The number is 5

Infinitive loops

- Can create infinitive loops by while and for statements
  - The loop is infinitive when the loop condition is always true

```
while (1)
{
   . . .
}

for (;;)
{
 . . .
}
```

`Break` statement

- Use break in an infinitive loop to terminate the loop
- Often use in infinitive loops

```
for (;;)
{
   …
   if (<condition>) break;
}
```

Example

```
#include <stdio.h>
int main()
{
   int n;
   while (1) {
   printf("Input a positive number: ");
   fflush(stdin);
   scanf("%d", &n);
   if (n<0 ){
   printf("Must be positive. Try again\n");}
   else
   break;
   }
   printf("The number is %d\n", n);
return 0;
}
```

Result

Input a positive number: -2

Must be positive. Try again

Input a positive number: 5

The number is 5

`Continue` statement

- When a continue statement is encountered, a loop will stop whatever it is doing and will go straight to the start of the next loop pass.

```
for (…)
{
   …
   if (< condition >) continue;
   …
}
```

Example

```
#include <stdio.h>
int main()
{
   int n;
   do {
   printf("Input a positive number: ");
   fflush(stdin);
   scanf("%d", &n);
   if (n < 0) {
   printf("Bad number. Try again\n");
   continue;
   }
   printf("The number is %d\n", n);
   }
   while (n<0);
return 0;
}
```

Result

Input a positive number: -2

Bad number. Try again

The number is 2

Input a positive number: 5

The number is 5

Exercises

(i) Write a program which prints out the prime factorization of a number (treat 2 as the first prime). For example,

- on input 6, desired output is: 2 3
- " " 24, " " : 2 2 2 3
- " " 23, " " : 23

(ii) Write a program which prints the first prime number greater than a number N on input.

(iii) Write a program to calculate the number PI using the following formula:PI/4 = 1-1/3+1/5 …. + (-1)n 1/(2n+1)

Solution (exercise 1)

```
factor = 2;
do
{
   if  (n%factor==0) {
      printf("%5d", factor);
      n = n / factor;
   } else {
      factor++;
   }
} while (n>1)
```

Solution (exercise 2)

```
first = n+1;
while(1) {
   isPrime = 1;
   for (factor = 2; factor<first; factor++)
     if (first%factor==0) {
       isPrime = 0;
       break;
     }
   if (isPrime) {
     printf("The first greater prime is %d", first);
     break;
   }
   first++;
}
```