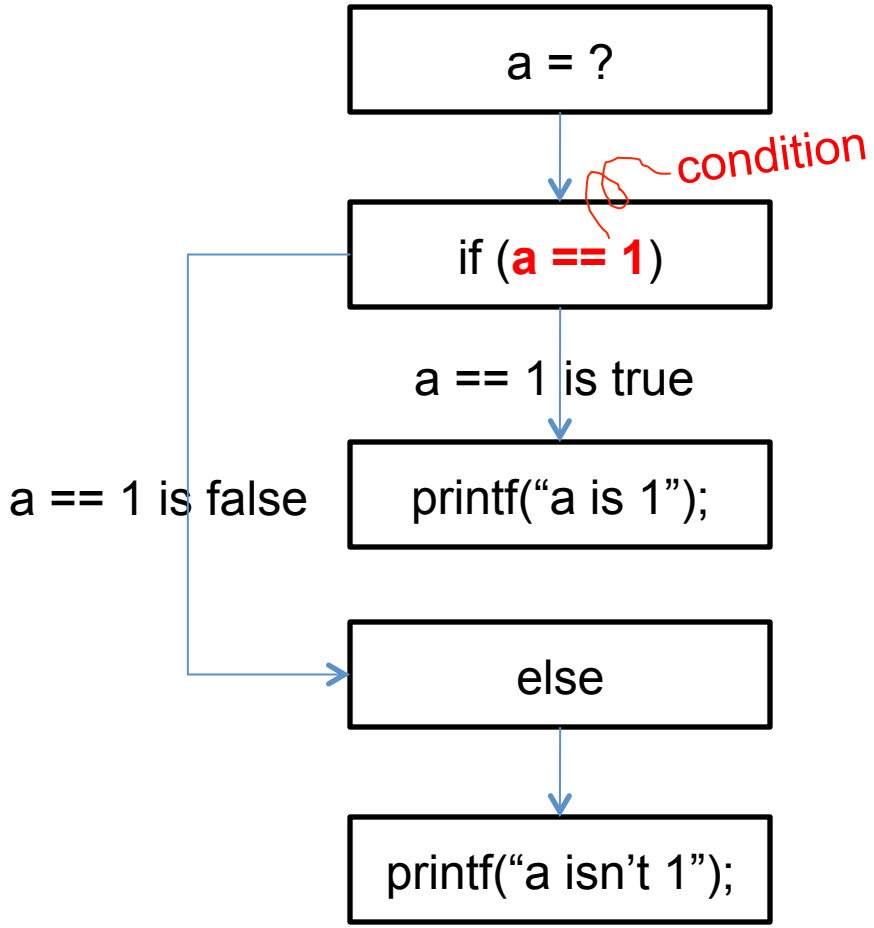
Basis and Practice in Programming

LAB4

If Statement (1/2)

- If-else statement
  - Control flow statement
  - Controlled by given condition
- Nested if statement
  - If in another if scope

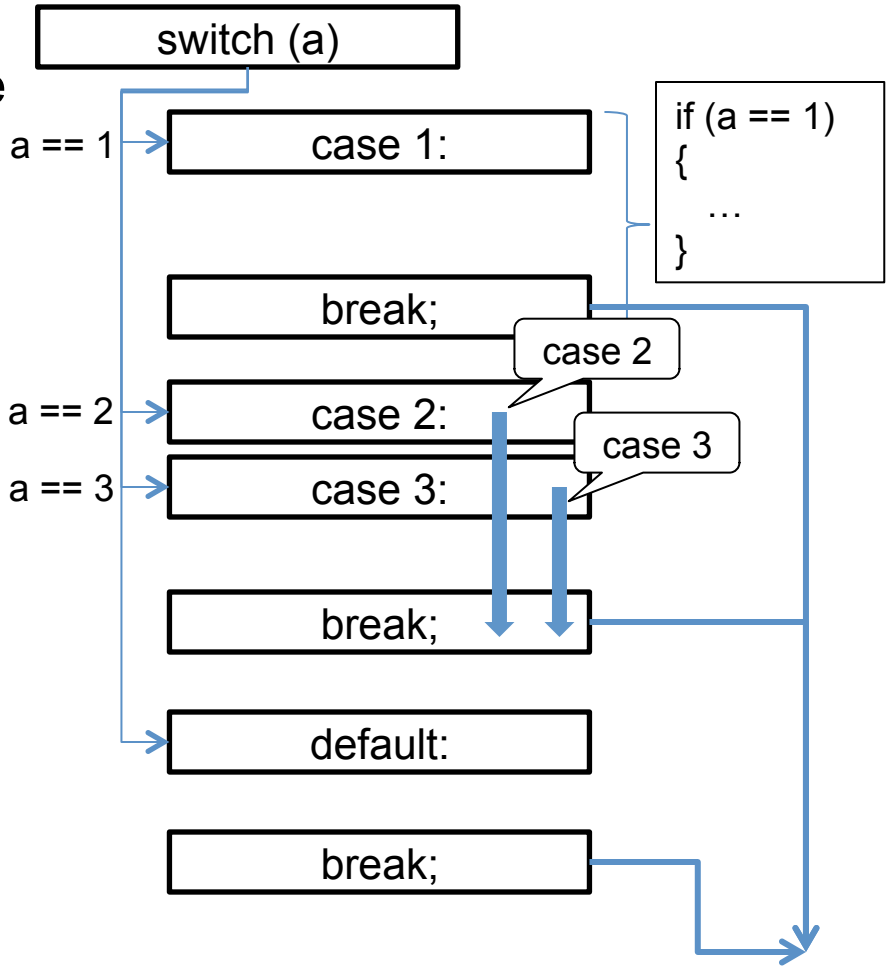


If Statement (2/2)

```
/* practice 1 : nested if statement exercise*/
#include <stdio.h>

int main(void)
{
    int a;
    scanf("%d", &a);

    if (a > 10) {
        if (a < 20) {
            printf("a is bigger than 10 and smaller than 20\n");
        } else {
            printf("a is bigger than 20\n");
        }
    } else {
        printf("a is smaller than 10\n"); // condition is false
    }

    return 0;
}
```

Switch statement

- Switch – Case
  - Each case statements must be ended by break
  - Default statement is executed when no matched case is in s witch context



Switch statement

- Switch – Case

```
/* practice 2 : if to switch */
#include <stdio.h>

int main(void)
{
    char c;
    scanf("%c", &c);

    if (c == 'm' || c == 'M')
        printf("Morning\n");
    else if (c == 'a' || c == 'A')
        printf("Afternoon\n");
    else if (c == 'e' || c == 'E')
        printf("Evening\n");
    else
        printf("Error\n");

    return 0;
}
```

```
/* practice 3 : switch */
#include <stdio.h>

int main(void)
{
    char c;
    scanf("%c", &c);

    switch(c) {
    case 'm':
    case 'M':
        printf("Morning\n"); break;
    case 'a':
    case 'A':
        printf("Afternoon\n"); break;
    case 'e':
    case 'E':
        printf("Evening\n"); break;
    default:
        printf("Error\n");
    }

    return 0;
}
```

Loop Statements

- Loop
  - Block for repeating some statements
  - Several statements
    - while, do-while, for
  - Ended by condition check

Loop Statements

- While
- Do-while
- For

```
while ( condition )
{
    //…
}
```

```
do
{
    //…
} while ( condition )
```

```
for (init ; end ; ops)
{
    //…
}
```

While

- While

```
/* practice 4 : while loop */
#include <stdio.h>

int main(void) {
   int num = 0;

   while ( num < 10 ) {
      printf (“num : %d\n”, num);
      num++;
   }

   return 0;
}
```

```
/* practice 5 : nested while loop */
#include <stdio.h>

int main(void) {
   int base = 1, multiplier;

   while ( base < 10 ) {
      multiplier = 1;
      while ( multiplier < 10 ) {
         printf(“%d * %d = %d ”, base,
               multiplier, base * multiplier);
         multiplier++;
      }
      printf (“\n”);
      base++;
   }

   return 0;
}
```

Do-While

- Do-While

```
/* practice 5 : do-while loop */
#include <stdio.h>

int main(void) {
   int num = 0;

   do {
      printf (“num : %d\n”, num);
      num++;
   } while (num < 10);

   return 0;
}
```

For

- For

```
/* practice 3 : while loop */
#include <stdio.h>

int main(void) {
   int num = 0;

   while ( num < 10 ) {
      printf (“num : %d\n”, num);
      num++;
   }

   return 0;
}
```

```
/* practice 6 : for loop */
#include <stdio.h>

int main(void) {
   int num;

   for (num=0; num<10; num++) {
      printf (“num : %d\n”, num);
   }

   return 0;
}
```

For

- Nested for statement

```
/* practice 7 : nested for loop */
#include <stdio.h>

int main(void) {
   int base, multiplier;

   for (base=1; base<10; base++) {
      for (multiplier=1;multiplier<10;multiplier++) {
         printf (“%d * %d = %d”, base, multiplier,
               base * multiplier);
      }
      printf(“\n”);
   }

   return 0;
}
```

Mixed Loop

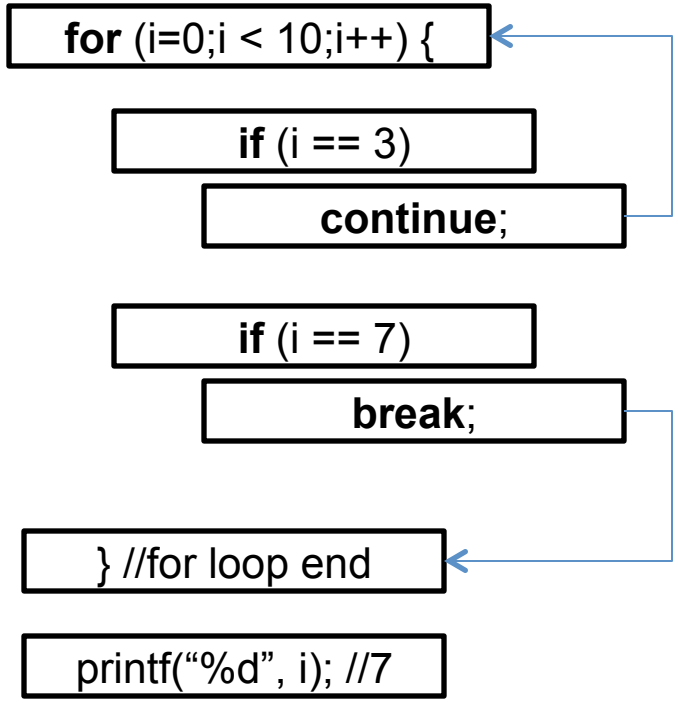
- Mixing several loop statements

```
/* practice 8 : mixed loop */
#include <stdio.h>

int main(void) {
   int base, multiplier;

   for (base=1; base<10; base++) {
      multiplier = 1;
      while ( multiplier<10 ) {
         printf ("%d * %d = %d", base, multiplier,
               base * multiplier);
         multiplier++;
      }
      printf("\n");
   }
```

```
   // … continued
   return 0;
}
```

Break/Continue

- Break
  - Used to escape from loop
- Continue
  - Used to skip this iteration



Break/Continue

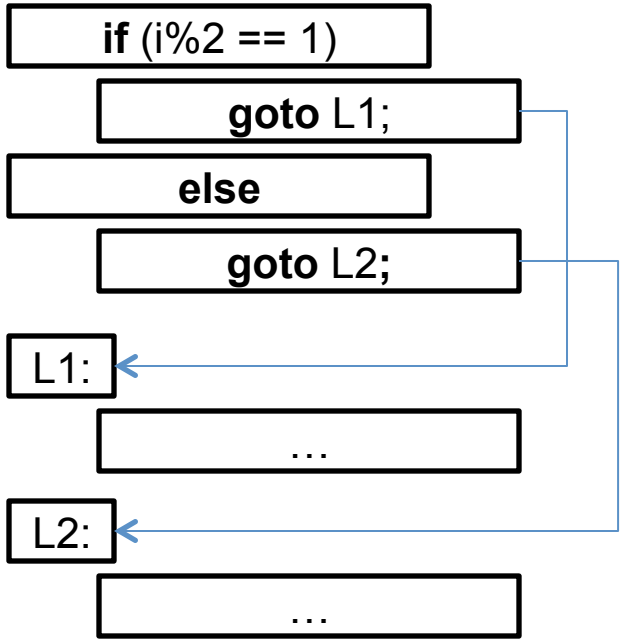
```
/* practice 9 : break and continue
#include <stdio.h>

int main(void)
{
    int i;

    for (i = 0;i < 10;i++) {
        if (i == 5) {
            printf (“continue\n”);
            continue;
        }

        printf(“%d\’s iteration\n”, i);

        if (i == 8) {
            printf (“break\n”);
            break;
        }
    }
    printf (“last i\’s value is %d\n”, i);

    return 0;
}
```

```
/* practice10 : break and continue
#include <stdio.h>

int main(void)
{
    int i = 0;

    for (;;) {
        i++;

        if (i > 10)
            break;

        continue;
        printf(“%d\’s iteration\n”, i);
    }

    printf (“last i\’s value is %d\n”, i);

    return 0;
}
```

Goto statement

- Goto statement
  - Unconditional jump to “label”
  - Because of program’s complexity, this statement is not recommended

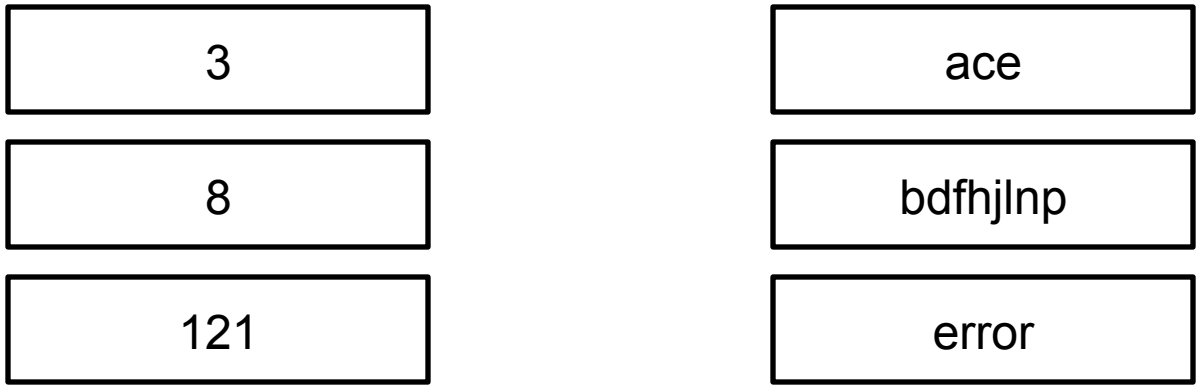
```
/* practice 11 : goto statement
#include <stdio.h>

int main(void)
{
    int i = 0;

loop:
    printf (“%d\’s iteration\n”, i);
    i++;

    if (i < 10)
        goto loop;

    return 0;
}
```



Exercise

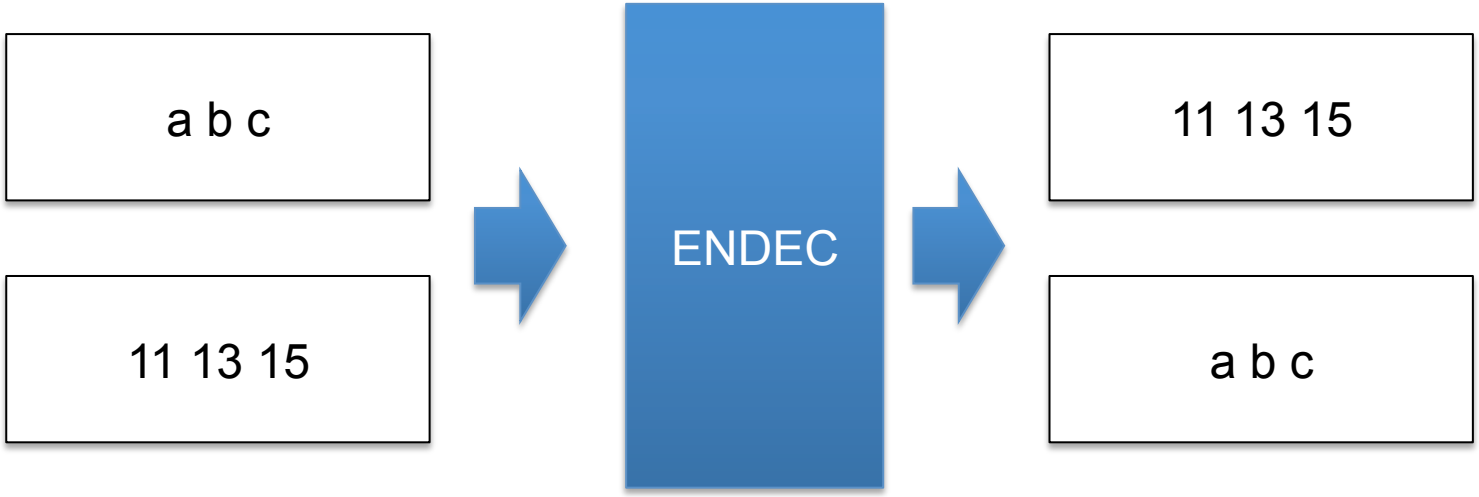
- Alphabet print
  - Taking a number as a input
  - Printing alphabets as much as input number
  - If input is odd number, prints alphabet on odd number position
  - If input is even number, prints alphabet on even numb er position
  - If input is bigger than printable alphabet length (>13), p rint “error”

Exercise



Homework#1 (due: ~ 10/26 11:59)

- ENDEC
  - Implementing encoder/decoder
  - If input is a character, print encoded code
  - If input is an encoded code, print decoded character



Homework#1 (due: ~ 10/27)

- ENDEC (continued)
  - Each character is separated by white space
  - If decoded codes are same to input characters, you ar e failed
  - MUST SUBMIT YOUR CODE BEFORE DEAD LINE