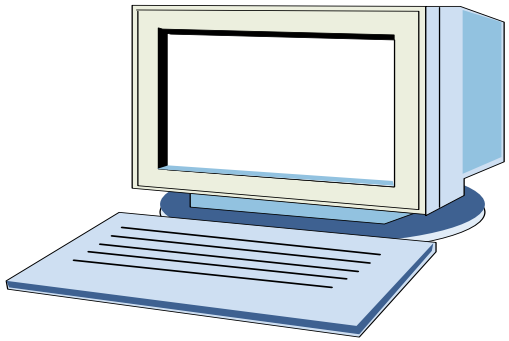
Console Interface

[ECE10002] C Programming

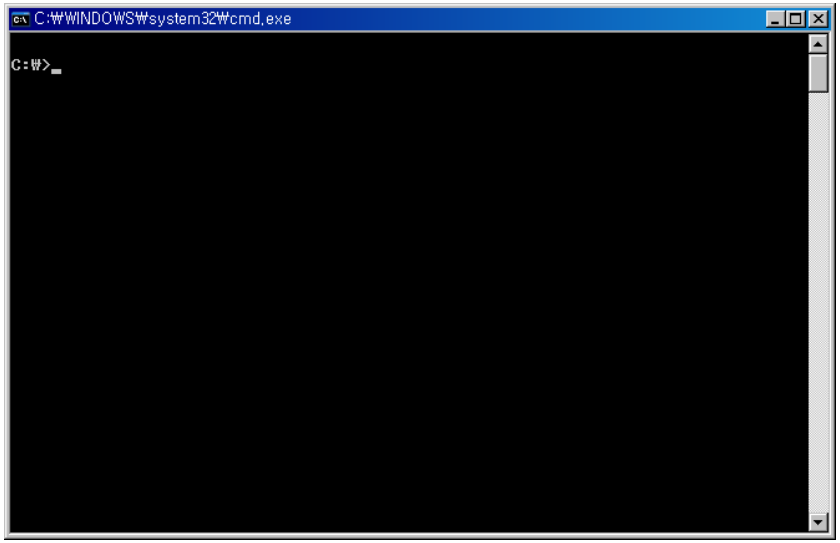
What is Consol ?

- Text console: text entry and display device
  - Text display + keyboard

  Ex) dummy terminal, console window



dummy terminal



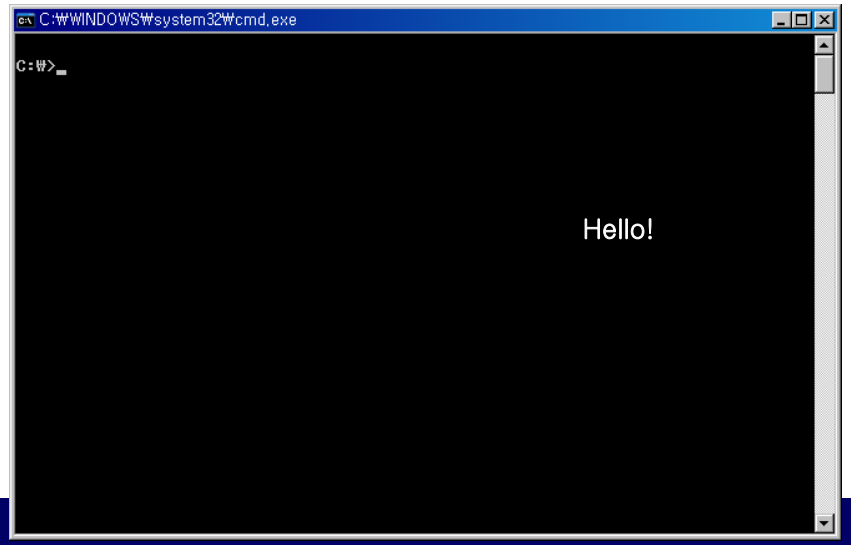
Console Interface

- Text I/O in C language standard
  - Input: scanf, getchar, gets, …
  - Output: printf, putchar, puts, …

→ Insufficient to make user-friendly interface.

Ex) printing a message at an arbitrary coordinate (x, y)

clear screen



Non-Standard Console Interface

- Many OS's, compliers, libraries provides non-standard functions required to implement user-friendly interface
  - Non-standard interface methods are vary with systems
- Examples of non-standard console interface
  - Display a message at an arbitrary position
  - Keyboard input without [Enter]
  - Non-blocking keyboard input

Console Interface for Dev C++

- Visual Studio and Dev-C++ allows to implement console interface using Window API (win32)
  - Documents about win32 console functions: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/console_functions.asp
- Include files
  - windows.h
  - or conio.h

Console Interface

- Output
  - Clear screen
  - Specifying coordinate to print message
- Input
  - Reading a character without enter
  - Reading a character without blocking

Console Output

- Clear screen

```
void clrscr(void)
{
   COORD Cur= {0, 0};
   unsigned long dwLen;
   FillConsoleOutputCharacter(GetStdHandle(STD_OUTPUT_HANDLE) , ' ',
    80*25, Cur, &dwLen);
   gotoxy(1, 1);
}
```

- Specifying coordinate to print message

```
void gotoxy(int x, int y)
{
   COORD Pos = {x - 1, y - 1};

   SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), Pos);
}
```

Console Output Example

- Example

```
#include <stdio.h>

#include <windows.h>
// or #include <conio.h>

void clrscr(void);
void gotoxy(int x, int y);

int main()
{
    int i = 0, j = 0;
    // fill screen with @'s
    for(i = 0; i < 25; i++){
        for(j = 0; j < 80; j++)
            printf("@");
    }
    printf("₩nPress Enter");
    getchar();

    clrscr();                    // clear screen
    for(i = 1; i < 20; i++){
        gotoxy(i, i);  // specify coordinate to print '*'
        printf("*");
    }

    gotoxy(1, 24);
    printf("₩nPress Enter to terminate");
    getchar();

    return 0;
}
```

Console Input

- Standard input functions in C language
  - scanf("%c", &c);
  - c = getchar();

  → Waits for [Enter]
    - Not sufficient for game programming

- Reading a character without enter
  - int getch(); // declared in windows.h or conio.h
    - Get a key without waiting for [Enter]
    - If a key input is waiting in a key buffer, getch() extracts it without blocking

getch() Example

- getch_example.c

```
#include <stdio.h>
#include <windows.h>                // conio.h on some compilers

int main()
{
    int c = 0;

    printf("Input a character (getchar): ");
    c = getchar();
    printf("c = [%c]₩n", c);
    getchar(); // just to consume [enter]

    printf("₩n");
    printf("Input a character (getch): ");
    c = getch();
    printf("₩nc = [%c]₩n", c);
}
```

Non-Blocking Key Input

- A non-blocking key input problem
  - Print natural numbers starting from 1, until 't' is pressed (run)

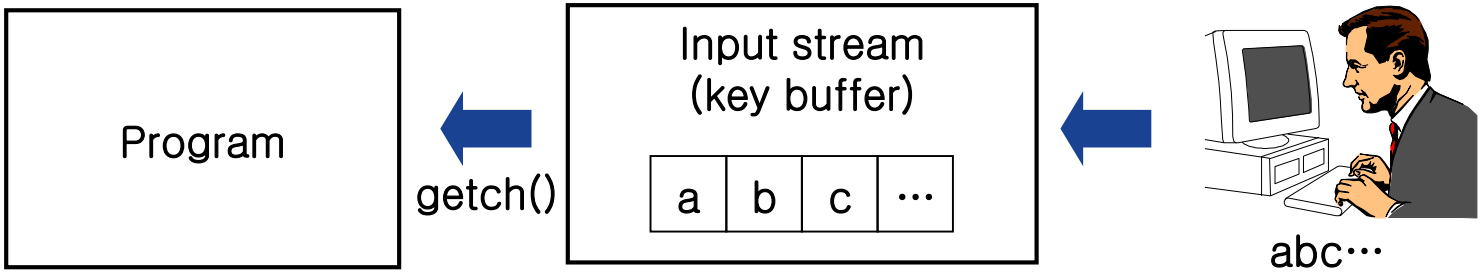
- Incorrect solution

```
void BlockingInput()
{
    int i = 1;
    char c = 0;

    while(c != 't'){
       printf("%d ", i++);
       c = getch();      // or getchar()
    }
}
```

Behavior of getch()

- Key buffer

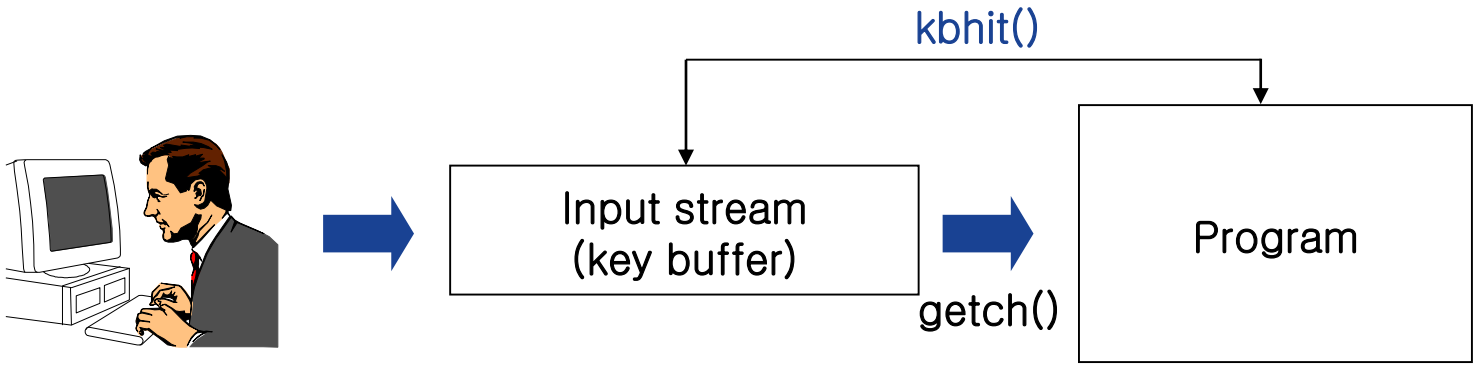


- Behavior of getch()
  - If key buffer is empty. (The user didn't press any key.)
    - getch() waits the user to press a key.

    → execution is blocked
  - If key buffer is not empty. (The user pressed a key.)
    - getch() returns the code of the pressed key without blocking
- How can we avoid blocking?
  - Let's call getch() only if key buffer is not empty.

Non-Blocking Key Input

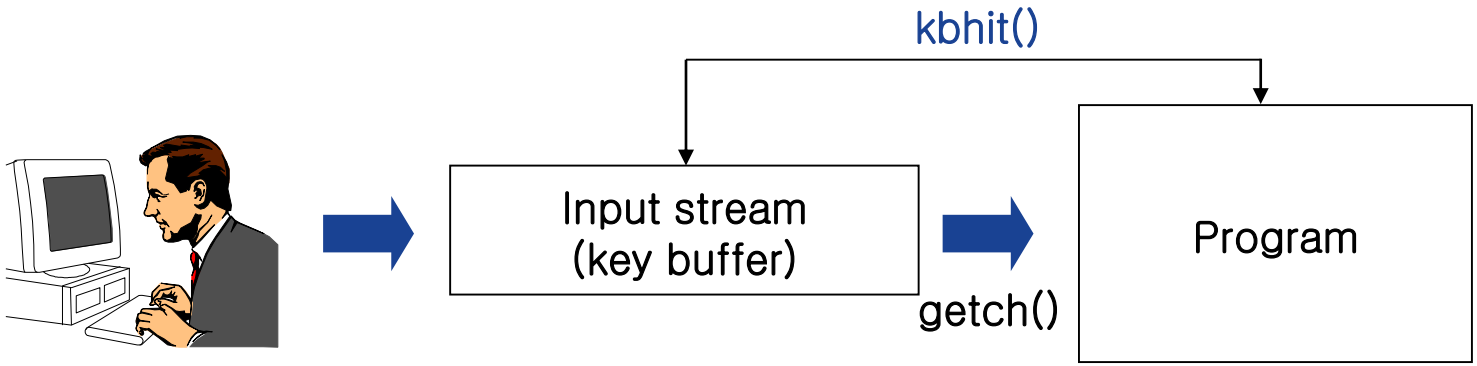
- Checking whether a key is pressed (checking key buffer)
  - Syntax: int kbhit(); // declared in conio.h or windows.h
  - Return value
    - 1 if a key is pressed
    - 0 otherwise



Non-Blocking Key Input

- Non-blocking key input

```
if(kbhit()){            // if a key is pressed
    c = getch();        //    get the pressed key (w/o blocking)
    // appropriate actions
}
// otherwise, continue execution
```



Non-Blocking Key Input

- Correct solution

```
void NonblockingInput()
{
    int i = 1;
    char c = 0;

    while(c != 't'){
       printf("%d ", i++);
       if(kbhit())          // check if a key is pressed
            c = getch();
    }
}
```

Delay

- Motivation: some programs run too fast
- Delay function
  - void sleep(unsigned long period); // UNIX
    - period: seconds or milliseconds
  - void Sleep(unsigned long period); // Windows
    - period: milliseconds

Ex)
```
void NonblockingInput()
{
    int i = 1;
    char c = 0;

    while(c != 't'){
       printf("%d₩n", i++);
       if(kbhit())
            c = getch();
       Sleep(200);                    // 0.2 second delay
    }
}
```