Hangman in C

Sept 15

HW1

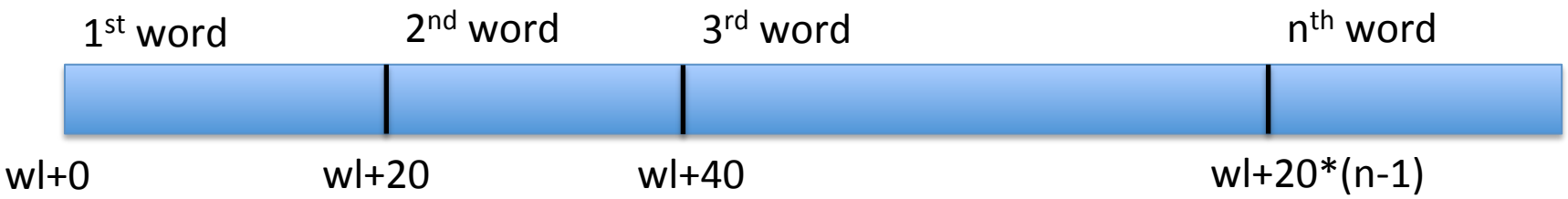
```
int main( void )
{
	char word_list[10][20];

	int word_cnt;

	printf("How many words do you want?:");
	scanf("%d", &word_cnt);

	printf("Please enter %d words\n", word_cnt);

	for( int i=0; i<word_cnt; i++) {
		printf("Enter %d th word:", i+1);
		scanf("%s", word_list[i]);
	}

    printf("## Word List ##\n");

	for( int i=0; i<10; i++) {
		printf("The %d th word is %s\n", i+1, word_list[i]);
	}

    system("pause");

    return 0;
}
```

HW1-solution

- Problem 1
  - word_cnt <10일 때, 입력된 단어들 외의 내용이 출력됨
  - Sol: 10 → word_cnt
- Problem 2
  - word_cnt > 10일 때, 준비된 단어 버퍼는 10개이므로 메모리 문제
  - Sol1: 최대 10개만 입력하게 함
  - Sol2: Dynamic allocation

Dynamic Word List

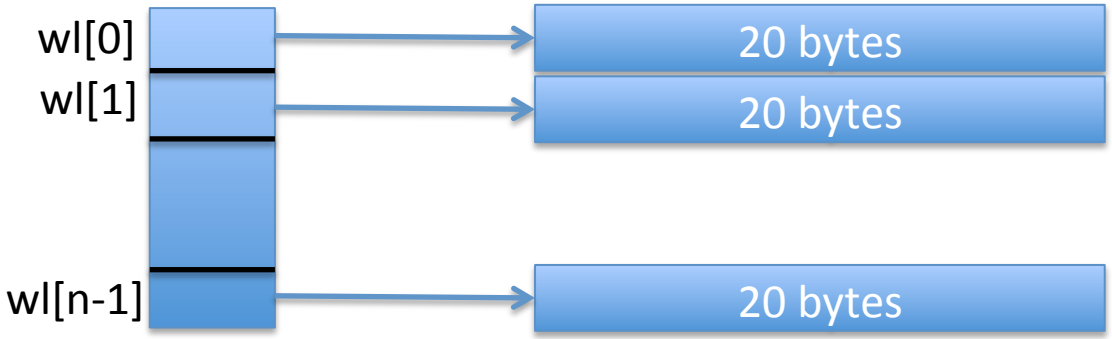
- Method 1
  - 20*단어갯수 byte의 char array 생성



  - wl = malloc(20*wordcnt);
  - index i인 단어의 시작 포인터: wl+20*I
  - free(wl);

Dynamic Word List

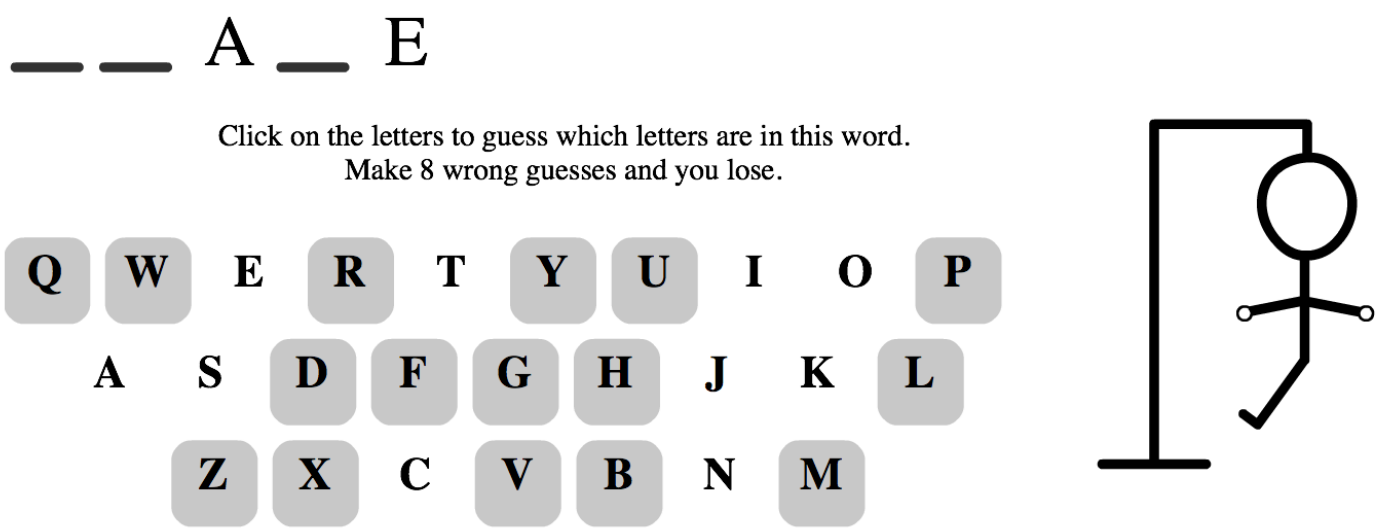
- Method 2
  - 단어갯수 만큼의 char * array 생성
  - 각 char *에 메모리 할당



  - wl = malloc(wordcnt * sizeof(char*));
  - wl[i] = malloc(20);
  - free(wl[i]); free(wl);

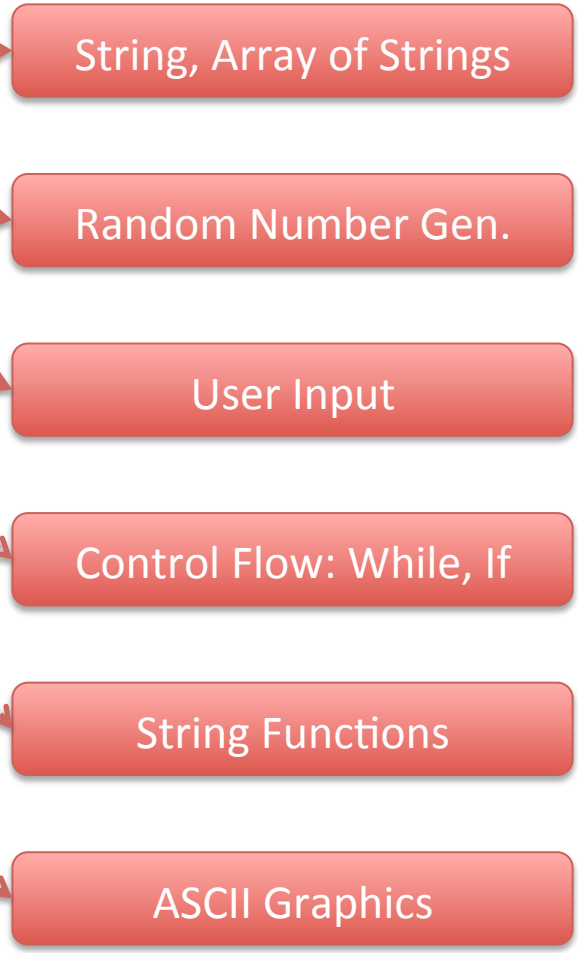
Hangman

- 컴퓨터가 단어를 하나 생각하면 그 단어에 포함되어 있는 알파벳을 하나씩 추측해서 전체 단어를 맞추는 게임이다.
- 단어에 없는 알파벳을 추측할 때마다 벌점이 추가되어, 단어를 마추기 전에 최대 벌점에 이르면 사람이 교수형에 매달리는 게임.



Program Flow

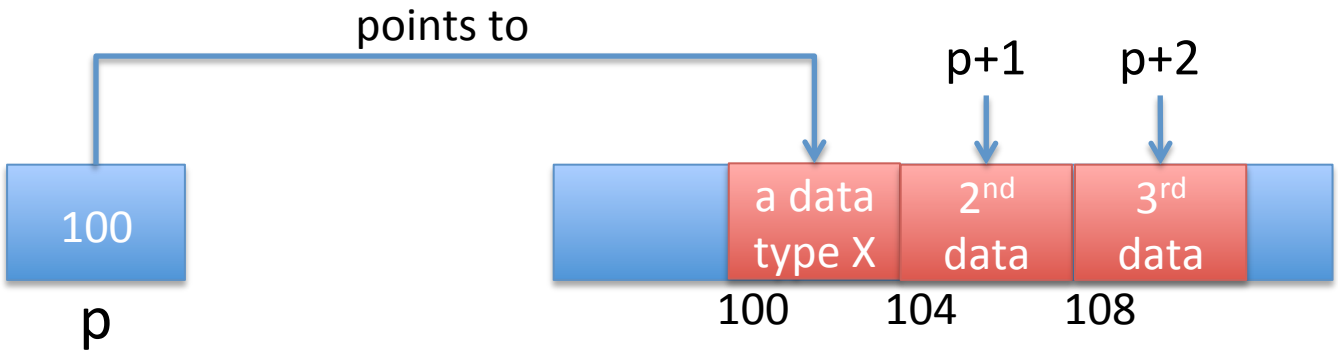
1. 단어모음 준비
2. 무작위 단어선택
3. 사용자가 글자 입력
    1. 이미 입력한 글자면 반복 3
4. 글자가 단어에 있으면
    1. 맞춘글자 표시, 반복 3
    2. 단어를 맞췄으면
        1. 끝 (WIN)
5. 없으면
    1. 틀린횟수 증가, 그림수정
    2. 최대 틀린횟수 도달했으면
        1. 끝 (LOSE)
6. 반복 3



Random Number Generation

- No true random number exists in Computer
  - 항상 같은 숫자가 같은 순서로 나옴
- int rand(): 0 ~ RAND_MAX 숫자 하나 리턴
- seed: 다른 숫자모음을 결정해줌
  - void srand(unsigned int seed);
  - srand(19);
- seed의 랜덤(?) 선택 방법: 시간을 이용
  - time_t t;
  - srand(time(&t));
  - time()은 Epoch time 리턴
- 0~N 숫자중 무작위 선택방법
  - rand() % N

Pointer & Array



- Pointer = (data 시작주소, data type)
- 변수 선언: <type>* <pointer>
  - int *p; char *q; int **x;
- 가리키는 data 읽기: *<pointer>
  - a = *p;
- 가리키는 위치 변경: <pointer> + n
  - <pointer>가 가리키는 data의 n번째 뒤에 있는 data를 가리키는 포인터
- 배열과의 관계
  - p[i] = *(p+i), 즉 p로부터 i번째 있는 데이터 내용

User Input

- getchar()
  - c = getchar();
- scanf()
  - scanf(“%s”, &str);

String Functions

- **strcpy(s1, s2);**
  - Copies string s2 into string s1.
- **strcat(s1, s2);**
  - Concatenates string s2 onto the end of string s1.
- **strlen(s1);**
  - Returns the length of string s1.
- **strcmp(s1, s2);**
  - Returns 0 if s1 and s2 are the same; less than 0 if s1<s2; greater than 0 if s1>s2.
- **strchr(s1, ch);**
  - Returns a pointer to the first occurrence of character ch in string s1.
- **strstr(s1, s2);**
  - Returns a pointer to the first occurrence of string s2 in string s1.

Game Loop

```
while(not lost yet && not won yet) {
        print incorrect letters left, used letters
        while(true) {
                get a letter
                break if new letter
        }
        add to used letters
        if( hit ) {
                update hit letters
        else
                wrong count++
        print hanger
}
if( lost ) …
if( won ) …
```