부록 2. 기본 클래스

I. Serial

1. if(Serial)

직렬 포트가 준비되었는지 검사하여 준비되었으면 true를 그렇지 않으면 false를 반환한다. Arduino UNO에서는 사용되지 않으며 Arduino Leonardo와 같이 가상의 시리얼 포트를 사용하는 경우 포트의 생성을 대기하기 위해 사용된다.

2. available

```
int available(void)
  반환값 : 직렬 포트 수신 데이터 버퍼에 저장된 데이터의 바이트 수
```

직렬 포트에 수신되어 버퍼에 저장되어 있는 데이터의 바이트 수를 반환한다. 버퍼에는 64바이트까지 저장될 수 있다.

3. begin

```
void begin(unsigned long baud)
void begin(unsigned long baud, byte config)
    baud : 속도
    config : 데이터 비트 수, 패리티, 정지 비트 설정
  반환값 : 없음
```

직렬 통신을 위한 전송 속도와 옵션을 설정한다. 전송 속도는 보율(baud rate)로 지정한다. 옵션으로 데이터 비트 수, 패리티 비트, 정지 비트 등을 설정할 수 있다. 디폴트값은 8 데이터 비트, 패리티 비트 없음, 1 정지 비트로 설정되어 있다.

4. end

```
void end(void)
  반환값 : 없음
```

직렬 통신을 종료한다.

5. find

```
bool find(char *target)
    target : 탐색 문자열
  반환값 : 발견 여부
```

직렬 통신 수신 버퍼를 읽어 주어진 문자열(target)이 발견되면 true를 반환하고 검색 시간이 초과되면 false를 반환한다. 검색 시간의 디폴트값은 1초이다.

6. findUntil

```
bool findUntil(char *target, char *terminator)
    target : 탐색 문자열
    terminator : 종료 문자열
  반환값 : 발견 여부
```

직렬 통신 수신 버퍼를 읽어 주어진 문자열(target)이나 종료 문자열 terminator가 발견되면 true를 반환하고 검색 시간이 초과되면 false를 반환한다. 검색 시간의 디폴트값은 1초이다.

7. flush

```
void flush(void)
  반환값 : 없음
```

직렬 통신 송신 버퍼에 있는 데이터의 전송이 완료될 때까지 대기한다.

8. parseFloat

```
float parseFloat(void)
  반환값 : 수신 버퍼에서 발견된 첫 번째 유효한 float 형 데이터
```

직렬 통신 수신 버퍼에서 첫 번째 유효한 실수를 반환한다. 수신 버퍼가 비어있거나 버퍼 끝까지 실수가 발견되지 않으면 0을 반환한다.

8. parseInt

```
int parseInt(void)
  반환값 : 수신 버퍼에서 발견된 첫 번째 유효한 int 형 데이터
```

직렬 통신 수신 버퍼에서 첫 번째 유효한 정수를 반환한다. 수신 버퍼가 비어있거나 버퍼 끝까지 정수가 발견되지 않으면 0을 반환한다.

9. peek

```
int peek(void)
  반환값 : 직렬 통신 수신 버퍼의 첫 번째 바이트 데이터 또는 -1
```

직렬 통신 수신 버퍼의 첫 번째 바이트 데이터를 반환한다. read 함수와 달리 peek 함수는 수신 버퍼에서 반환한 데이터를 제거하지 않는다. 수신 버퍼가 비어있는 경우 -1을 반환한다.

10. print

```
size_t print(value, format)
    value : 출력값 (char, char 배열, String, 정수, 실수 등)
    format : 출력 형식
  반환값 : 직렬 포트로 출력된 바이트 수
```

주어진 값을 ASCII 형식으로 직렬 포트로 출력한다. 정수의 경우 출력 형식을 2진법(BIN), 8진법(OCT), 10진법(DEC), 16진법(HEX) 중 하나를 선택할 수 있고, 실수의 경우 소숫점 이하 자릿수를 지정할 수 있다. 디폴트값은 소수점 이하 2자리이다. 문자, 문자열, String 객체 등도 출력할 수 있다. 직렬 포트로 출력된 데이터의 바이트 수를 반환한다.

11. println

```
size_t println(value, format)
    value : 출력값 (char, char 배열, String, 정수, 실수 등)
    format : 출력 형식
  반환값 : 직렬 포트로 출력된 바이트 수
```

데이터 출력 후 개행 문자를 추가로 출력하는 점을 제외하면 print 함수와 동일하다.

12. read

```
int read(void)
  반환값 : 직렬 통신 수신 버퍼의 첫 번째 문자 또는 -1
```

직렬 통신 수신 버퍼에서 첫 번째 문자를 읽어 반환한다. 수신 버퍼가 비어 있으면 -1을 반환한다.

13. readBytes

```
size_t readBytes(char *buffer, size_t length)
    buffer : 입력 문자를 저장한 버퍼
    length : 입력 받을 최대 문자 수
  반환값 : 입력 받은 문자 수
```

직렬 통신 수신 버퍼의 데이터를 읽어 buffer에 저장한다. length로 지정한 바이트 수의 데이터를 읽었거나 시간 초과가 발생하면 읽어 들인 문자의 수를 반환한다.

14. readBytesUntil

```
size_t readBytesUntil(char terminator, char *buffer, size_t length)
    terminator : 종료 문자
    buffer : 입력 문자를 저장한 버퍼
    length : 입력 받을 최대 문자 수
  반환값 : 입력 받은 문자 수
```

수신 버퍼에서 종료 문자가 발견된 경우 반환한다는 점을 제외하면 readBytes 함수와 동일하

다.

15. setTimeout

```
void setTimeout(unsigned long timeout)
    timeout : 대기 시간
  반환값 : 없음
```

readBytes 함수나 readBytesUntil 함수의 대기 시간을 밀리초 단위로 설정한다. 디폴트값은 1초이다.

16. write

```
size_t wirte(uint8_t ch)
size_t write(const char *str)
size_t write(const uint8_t *buffer, size_t size)
    ch : 출력 문자
    str : 출력 문자열
    buffer : 출력 데이터열
    size : 버퍼의 크기
  반환값 : 출력한 바이트 수
```

주어진 값을 이진 형식으로 직렬 포트로 출력한다. 문자열의 경우 print 함수와 동일한 결과를 가져오지만 write(65)는 'A'가 출력되는 반면 print(65)는 '65'가 출력되는 점에서 다르다. 직렬 포트로 출력된 데이터의 바이트 수를 반환한다.

17. serialEvent

```
void serialEvent(void)
  반환값 : 함수
```

Serial 클래스의 멤버 함수가 아니며 수신 버퍼에 데이터가 수신된 경우 호출되는 함수이다. serialEvent 함수는 그 이름과 다르게 인터럽트 방식이 아닌 폴링 방식으로 동작한다.

II. String

1. String

```
String(value)
String(value, base)
```

String 클래스의 생성자로 정수, 문자, 문자 배열, String 객체 등 다양한 데이터를 지정할 수 있다. 정수가 주어진 경우 base 값을 통해 진법을 지정할 수 있으며 디폴트값은 10이다. 단, 실수값은 String 객체로 변환할 수 없다.

2. charAt

```
char charAt(unsigned int index)
    index : 문자열에서 찾고 싶은 문자의 위치로 0부터 시작된다.
  반환값 : 문자열에서 index 번째 문자
```

문자열 내에서 지정한 위치를 문자를 반환한다. 연산자 '[]'와 동일한 기능을 수행한다.

3. compareTo

```
int compareTo(const String &string2)
    string2 : 비교 대상이 되는 문자열
  반환값 : 순서상으로 string2가 먼저 나오면 양의 값, string2와 동일하면 0, string2가
           뒤에 나오면 음의 값을 반환
```

두 문자열을 사전 배열 순서에 따라 비교하여 비교한 결과를 반환한다.

4. concat

```
unsigned char concat(const String &str)
unsigned char concat(int num)
    str : 이어 붙일 문자열
    num : 이어 붙일 숫자. 문자열로 변환되어 이어 붙인다.
  반환값 : 성공하면 1, 실패하면 0을 반환
```

문자열 뒤에 인자로 주어진 문자열을 이어 붙인다. 성공하면 1을 실패하면 0을 반환한다. 연산자 '+'와 동일한 기능을 수행한다. 이외에도 문자형, long 형 등을 매개변수로 가지는 여러 함수들이 다중 정의되어 있다.

5. endsWith

```
unsigned char endsWith(const String &str)
    str : 비교 대상 문자열
  반환값 : 문자열이 str로 끝나는 경우 1, 아닌 경우 0을 반환
```

문자열이 str에 주어진 문자열로 끝나는지 검사하여 str로 끝나면 1을, str로 끝나지 않으면 0을 반환한다.

6. equals

```
unsigned char equals(const String &str)
    str : 비교 대상 문자열
  반환값 : 문자열이 str과 동일한 경우 1, 아닌 경우 0을 반환
```

문자열이 str에 주어진 문자열과 동일한지 비교하여 동일하면 1을, 동일하지 않으면 0을 반환한다.

7. equalsIgnoreCase

```
unsigned char equalsIgnoreCase(const String &str)
    str : 비교 대상 문자열
  반환값 : 문자열이 str과 동일한 경우 1, 아닌 경우 0을 반환
```

문자 비교 시 대소문자를 동일한 문자로 간주하는 것을 제외하면 equals와 동일하다.

8. getBytes

```
void getBytes(unsigned char *buf, unsigned int bufsize)
    buf : 문자들을 복사할 버퍼
    bufsize : 버퍼의 크기
  반환값 : 없음
```

String 객체의 문자들을 (bufsize - 1)개 이내에서 버퍼로 복사한다.

9. indexOf

```
int indexOf(char ch)
int indexOf(char ch, unsigned int fromIndex)
int indexOf(const String &str)
int indexOf(const String &str, unsigned int fromIndex)
    ch, str : 탐색할 문자 또는 문자열
    fromIndex : 탐색을 시작할 위치
  반환값 : 검색 문자 또는 문자열이 처음 발견된 위치를 반환하며 발견되지 않으면 -1을
           반환
```

문자열 내에서 주어진 문자나 문자열을 검사하여 첫 번째 발견된 위치를 반환하며 존재하지 않는 경우 -1을 반환한다. fromIndex 값으로 시작 위치를 지정할 수 있다.

10. lastIndexOf

```
int lastIndexOf(char ch) const
int lastIndexOf(char ch, unsigned int fromIndex) const
int lastIndexOf(const String &str) const
int lastIndexOf(const String &str, unsigned int fromIndex) const
    ch, str : 탐색할 문자 또는 문자열
    fromIndex : 탐색을 시작할 위치
  반환값 : 검색 문자 또는 문자열이 발견된 첫 번째 위치를 반환하며 발견되지 않으면
           -1을 반환
```

문자열의 끝에서부터 역방향으로 검사하는 것을 제외하면 indexOf 함수와 동일하다.

11. length

```
unsigned int length(void)
  반환값 : 문자열의 길이
```

문자열의 길이를 반환한다.

12. replace

```
void replace(char find, char replace)
void replace(const String& find, const String& replace)
    find : 찾을 문자열
    replace : 바꿀 문자열
  반환값 : 없음
```

find에 주어진 문자나 문자열을 찾아 replace에 주어진 문자나 문자열로 바꾼다.

13. setCharAt

```
void setCharAt(unsigned int index, char ch)
    index : 바꿀 문자의 위치
    ch : 바꿀 문자
  반환값 : 없음
```

문자열의 index번째 문자를 ch로 바꾼다. 대입 연산자의 좌변에 놓이는 연산자 '[]'와 동일한 기능을 수행한다.

14. startsWith

```
unsigned char startsWith(const String &str)
    str : 비교 대상 문자열
  반환값 : 문자열이 str로 시작하는 경우 1, 아닌 경우 0을 반환
```

문자열이 str에 주어진 문자열로 시작하는지 검사하여 str로 시작하면 1을, str로 시작하지 않으면 0을 반환한다.

15. substring

```
String substring(unsigned int from)
String substring(unsigned int from, unsigned int to)
    from : 추출할 부분 문자열의 시작 위치
    to : 추출할 부분 문자열의 끝 위치
  반환값 : 추출한 부분 문자열을 반환
```

문자열의 부분 문자열을 반환한다. from에 지정된 위치의 문자는 포함되는 반면 to 위치의 문자는 포함되지 않는다. 끝 위치가 지정되지 않은 경우에는 시작 위치에서 문자열의 끝까지 부분 문자열로 반환한다.

16. toCharArray

```
void toCharArray(char *buf, unsigned int bufsize)
    buf : 문자들을 복사할 버퍼
    bufsize : 버퍼의 크기
  반환값 : 없음
```

String 객체의 문자들을 (bufsize - 1)개 이내에서 버퍼로 복사한다.

17. toInt

```
long toInt(void)
  반환값 : 변환된 정수값을 반환
```

문자열을 정수로 변환하여 반환한다. 변환에 실패할 경우 0을 반환한다.

18. toLowerCase

```
void toLowerCase(void)
  반환값 : 없음
```

문자열 내의 대문자를 소문자로 바꾼다.

19. toUpperCase

```
void toUpperCase(void)
  반환값 : 없음
```

문자열 내의 소문자를 대문자로 바꾼다.

20. trim

```
void trim(void)
  반환값 : 없음
```

문자열의 시작과 끝 부분에 포함된 공백 문자(white space)를 제거한다. 공백 문자에는 single space (ASCII 32), tab (ASCII 9), vertical tab (ASCII 11), form feed (ASCII 12), carriage return (ASCII 13), newline (ASCII 10) 등이 포함된다.

21. [] (문자열 요소)

```
char operator [] (unsigned int index)
    index : 문자열에서 찾고 싶은 문자의 위치
  반환값 : 문자열에서 index 번째 문자
```

문자열 내에서 지정한 위치의 문자를 반환한다. charAt 함수와 동일한 기능을 수행한다.

22. + (문자열 연결)

```
String & operator += (const String &rhs)
String & operator += (const char *cstr)
String & operator += (char c)
String & operator += (unsigned char num)
String & operator += (int num)
String & operator += (unsigned int num)
String & operator += (long num)
String & operator += (unsigned long num)
    rhs, cstr, c, num : 연결할 문자, 문자열 또는 숫자
  반환값 : 문자열을 연결한 새로운 문자열
```

문자열을 연결하여 새로운 문자열을 반환한다.

23. == (문자열 비교)

```
unsigned char operator == (const String &rhs)
unsigned char operator == (const char *cstr)
    rhs, cstr : 비교 문자열
  반환값 : 비교 문자열과 같으면 1을, 다르면 0을 반환
```

두 문자열을 비교하여 동일하면 1을, 다르면 0을 반환한다. equals 함수와 동일한 기능을 수행한다.