【std::string】

std::string は C 言語の文字配列よりも簡単に文字列を扱えます。
使用するには「string ヘッダーファイル」のインクルードが必要です。

typedef basic_string<char> string;
basic_string は文字列を扱うクラスで、正確には std::string は basic_string を typedef したものです。

```
std::string str;                    //string 型の変数
str = "abc";                        //=演算子で文字列を代入できる
std::cin >> str;                    //入力した文字列を代入できる

str = "abc";
str = str + "def";                  //+演算子で文字列を結合できる
std::cout << str << std::endl;      //出力結果は「abcdef」

if (str == "abcdef") {}             //条件式での使用できる

int num = 10;                       //int 型の変数
str = "abc" + num;                  //文字列以外は結合できない
std::cout << str << std::endl;      //何も出力されない
```

＜文字配列との互換＞
同じ文字列を記録するものですが、文字配列（char 型配列）と string は異なるデータ型です。

```
std::string str = "abc";
printf("%s¥n", str);    //この行はエラーになる。
```

printf 関数の第 2 引数は char 型配列。変数 str は char 型配列ではありません。
このように C 言語の関数では char 型配列を使用しており、string は使用できません。
string を char 型配列に変換するには「c_str」を使用します。

```
printf("%s¥n", str.c_str());    //変数 str の値を char 型配列に変換して使用する。
```

【std::stringstream】

std::stringstream は std::string への文字列の入力を行うものです。
使用するには sstream ヘッダーファイルのインクルードが必要です。

typedef basic_stringstream<char> stringstream;
basic_stringstream は文字列ストリームを扱うクラスで、正確には std::stringstream は basic_stringstream を typedef したものです。

```
std::stringstream stream;           //stream 型の変数
stream << "abc";                     // <<演算子を使用して文字列を追加する
stream << "def";                     //「abc」に追加されて「abcdef」になる
stream << 10;                       //数値を追加する事もできる
std::string str = stream.str();     //「str」で string に変換できる
std::cout << str << std::endl;      //「abcdefg10」と出力

std::string str2 = "abc";
std::stringstream stream2(str2)     //「abc」を追加する

int no = 10;
std::string name = "A 太郎";
std::stringstream stream3;
stream3 << no << "," << name;
std::cout << stream3.str();         //「10,A 太郎」と出力
```

<<演算子は文字列の「代入」ではなく「追加」になり、使用する度に文字列が値の末尾に追加されていく仕様になっています。
変数 stream の値を消去したい場合は
stream.str("");
と記述します。