```
#include <iostream>
#include <algorithm>
#include <map>

std::map<char, char> generer_cle() {
	std::map<char, char> cle;
	std::string alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
	std::string alphabet_shuffle = alphabet;
	std::random_shuffle(alphabet_shuffle.begin(), alphabet_shuffle.end());

	for (size_t i = 0; i < alphabet.size(); ++i) {
		cle[alphabet[i]] = alphabet_shuffle[i];
	}

	return cle;
}

std::string chiffrement_monoalphabetique(const std::string& message, const std::map<char,
char>& cle) {
	std::string message_chiffre;
	for (char lettre : message) {
		if (isalpha(lettre)) {
			message_chiffre += cle.at(toupper(lettre));
		} else {
			message_chiffre += lettre;
		}
	}
	return message_chiffre;
}

std::string dechiffrement_monoalphabetique(const std::string& message_chiffre, const
std::map<char, char>& cle) {
	std::map<char, char> cle_inverse;
	for (const auto& pair : cle) {
		cle_inverse[pair.second] = pair.first;
	}

	std::string message_dechiffre;
	for (char lettre : message_chiffre) {
		if (isalpha(lettre)) {
			message_dechiffre += cle_inverse.at(lettre);
		} else {
			message_dechiffre += lettre;
		}
	}
	return message_dechiffre;
}

int main() {
	std::map<char, char> cle = generer_cle();
	std::string message_original = "Bonjour, ceci est un exemple.";
	std::string message_chiffre = chiffrement_monoalphabetique(message_original, cle);
	std::string message_dechiffre = dechiffrement_monoalphabetique(message_chiffre, cle);

	std::cout << "Message original: " << message_original << std::endl;
	std::cout << "Message chiffré: " << message_chiffre << std::endl;
```

```
    std::cout << "Message déchiffré: " << message_dechiffre << std::endl;

    return 0;
}
```