Problem Set

Multiple Choice

1. Which of the following strings is a valid method to convert a number to a string in Python?

   - a) `str(x)`
   - b) `int(x)`
   - c) `float(x)`
   - d) `bool(x)`

2. What is the result of the following code?

   ```
   s = 'spam' * 3
   print(s)
   ```

   - a) `spamspamspam`
   - b) `spam`
   - c) `spamspam`
   - d) `spamspamspamspam`

Long Answer

3. Explain the difference between string concatenation and string replication in Python.

4. Write a code snippet that uses the `join` method to concatenate the strings `"hello"`, `"world"`, and `"!"` with a space in between each word.

5. Write a code snippet that searches for the substring `"red"` in the string `"Fred"`. If the substring is found, print `"Fred is red"`, otherwise, print `"Fred is not red"`.

Solution Set

Multiple Choice

1. a) `str(x)` is the correct answer. The `str()` function in Python converts the given object into a string.

2. a) `spamspamspam` is the correct answer. The `*` operator in Python can be used for string replication. In this case, it repeats the string `"spam"` three times, resulting in `"spamspamspam"`.

Long Answer

3. In Python, string concatenation is the process of combining two or more strings into one. This can be done using the `+` operator. For example, `"hello" + "world"` would result in the string `"helloworld"`.

String replication, on the other hand, is the process of repeating a string multiple times. This can be done using the `*` operator. For example, `"spam" * 3` would result in the string `"spamspamspam"`.

4. ```
   words = ["hello", "world", "!"]
   concatenated_string = " ".join(words)
   print(concatenated_string)
   ```

   This code snippet uses the `join` method to concatenate the strings `"hello"`, `"world"`, and `"!"`. The `join` method takes a list of strings as an argument and concatenates them together using the specified separator (`" "` in this case). The result would be the string `"hello world !"`.

5. ```
   if "red" in "Fred":
       print("Fred is red")
   else:
       print("Fred is not red")
   ```

   This code snippet searches for the substring `"red"` in the string `"Fred"`. The `in` keyword is used to check if the substring is present in the string. If it is, it prints `"Fred is red"`. Otherwise, it prints `"Fred is not red"`. In this case, since the substring is present in the string, it would print `"Fred is red"`.