LAB 04 – JAVASCRIPT (CONT.)

I/. Javascript Validation Form

1/. Regular expression in Javascript

Javascript supports regular expressions through the RegEx library.

Regular expression works according to the matching rules, and gets the result of that match through the rules provided by the RegEx library. Based on these rules, programmers will write expressions and apply them to their problems.

RegEx (regular expression) is a set of rules, based on these rules we will write expressions that match between strings. This is an advanced library built into Javascript's RegExp object.

Syntax:

```
/pattern/modifiers
```

In there:

- pattern is a Regular Expression string
- modifiers is the configuration parameter for the pattern string, and it has the following values:
    - i : match regardless of case
    - g : match the entire search string
    - m : matches all newline data (multiline)

In JavaScript, regular expressions are often used with the two string methods: **search()** and **replace().**

- The search() method uses an expression to search for a match, and returns the position of the match.
- The replace() method returns a modified string where the pattern is replaced.

Regular Expression Patterns:

a/. Brackets are used to find a range of characters:

[abc]	Any of the characters a, b, or c
[A-Z]	Any character from uppercase A to uppercase Z
[a-z]	Any character from lowercase a to lowercase z
[A-z]	Any character from uppercase A to lowercase z

[^abc]	Not any of the characters a, b, or c
[^A-Z]	Not any character from uppercase A to uppercase Z
[^a-z]	Not any character from lowercase a to lowercase z
[^A-z]	Not any character from uppercase A to lowercase z

[0-9]	Find any character between the brackets (any digit)
[^0-9]	Find any character NOT between the brackets (any non-digit)
(x|y)	Find any of the alternatives specified

b/. Metacharacters are characters with a special meaning:

Metacharacter	Description
.	Find a single character, except newline or line terminator
\w	Find a word character
\W	Find a non-word character
\d	Find a digit
\D	Find a non-digit character
\s	Find a whitespace character
\S	Find a non-whitespace character
\b	Find a match at the beginning/end of a word, beginning like this: \bHI, end like this: HI\b
\B	Find a match, but not at the beginning/end of a word
\0	Find a NULL character
\n	Find a new line character
\f	Find a form feed character
\r	Find a carriage return character
\t	Find a tab character
\v	Find a vertical tab character
\xxx	Find the character specified by an octal number xxx
\xdd	Find the character specified by a hexadecimal number dd
\udddd	Find the Unicode character specified by a hexadecimal number dddd

c/. Quantifiers define quantities:

Quantifier	Description
n+	Matches any string that contains at least one *n*
n*	Matches any string that contains zero or more occurrences of *n*
n?	Matches any string that contains zero or one occurrences of *n*
n{X}	Matches any string that contains a sequence of *X* *n*'s
n{X,Y}	Matches any string that contains a sequence of X to Y *n*'s
n{X,}	Matches any string that contains a sequence of at least X *n*'s
n$	Matches any string with *n* at the end of it
^n	Matches any string with *n* at the beginning of it
?=n	Matches any string that is followed by a specific string *n*
?!n	Matches any string that is not followed by a specific string *n*

The structure of a string always has a start character and an end character. And in RegEx there are also some characters that help with the convention of where the string begins and where it ends. Its structure will be as follows:

```
/^pattern$/
```

Where the ^ character is the start of the string, and the $ is the end of the string.

In JavaScript, the RegExp object is a regular expression object with predefined properties and methods.

- Using test()

The test() method is a RegExp expression method. It searches a string for a pattern, and returns true or false, depending on the result.

- Using exec()

The exec() method is a RegExp expression method.

It searches a string for a specified pattern, and returns the found text as an object.

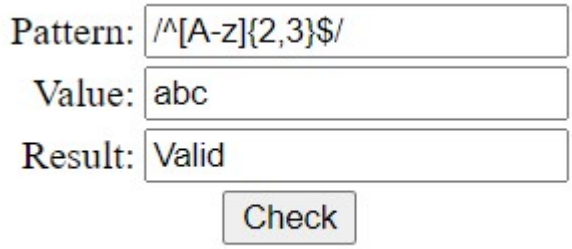
If no match is found, it returns an empty (null) object.

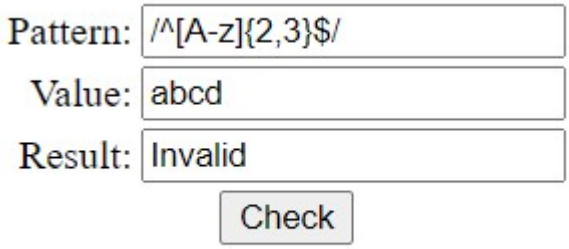
- Use the string.match function to get the entire regex string result.

Example 01:

Create a form for checking Regular Expression like this:

Test:


Pattern: /^[A-z]{2,3}$/
Value: abc
Result: Valid
Check


Pattern: /^[A-z]{2,3}$/
Value: abcd
Result: Invalid
Check

You can try:

```html
<form>
    <table>
        <tr>
            <td align="right">Pattern:</td>
            <td>
                <input type="text" id="pattern"/>
            </td>
        </tr>
        <tr>
            <td align="right">Value:</td>
            <td>
                <input type="text" id="value"/>
            </td>
        </tr>
        <tr>
            <td align="right">Result:</td>
            <td>
                <input type="text" id="result"/>
            </td>
        </tr>
        <tr>
            <td colspan="2" align="center">
                <input type="button" value="Check" onClick="check()"/>
            </td>
        </tr>
    </table>
</form>
```

```html
<script>
    function check() {
        try {
            var exp = RegExp(String(document.getElementById("pattern").value).slice(1,-1));
            document.getElementById("result").value = exp.test(document.getElementById("value").value) ?
                                                      "Valid" :
                                                      "Invalid";
        } catch (err) {
            document.getElementById("result").value = err.message;
        }
    }
</script>
```

Example 02: Write regular expression a/. Write a regular expression that checks for integers. b/. Write a regular

expression that checks for real numbers. c/. Write a regular

expression that checks the date in the format dd/mm/yyyy.

d/. Write a regular expression that checks landline numbers in the format (0710) 3777888. e/. Write a regular expression that checks for a cell phone number. f/. Write a regular expression that checks email.

You can try:

a/. p = /^[\+\-]?\d+$/ b/. p =

/^[\+\-]?\d+(\.\d+)?$/ c/. p =

/^\d{2}\/\d{2}\/\d{4}$/ d/. p

= /^\(0\d{1,3}\)\d{7}$/ e/.

p = /^(01\d{9}|09\d{8})$/ p =

/^(\(0\d{1,3}\)\d{7}|01\d{9}|09\d{8})$/ f/. p = /^[A-z0-9]+([\.\-

_]\w+)*@\w+([\.\-_]\w+)*(\.\w{2,4})+$/

2/. Form Validation

It is important to validate the form submitted by the user because it can have inappropriate values. So, validation is must to authenticate user.

JavaScript provides facility to validate the form on the client-side so data processing will be faster than server-side validation. Most of the web developers prefer JavaScript form validation.

Through JavaScript, we can validate name, password, email, date, mobile numbers and more fields.

Example:

I have a form like this:

I want to validate this form like this:

- User must enter Username before submit form. If username is not entered, an alert will show “Name can't be blank”.
- User must enter Password according to the rules: Password must be at least 6 characters long. If user enter password wrong, an alert will show “Password must be at least 6 characters long”.

And if you validate form successfully, an html page (called validate-success.html) will show result registration’s information. A page likes this:

How can you do that?

You can try by 2 ways:

- The first way: You can validate form by using alert dialog

+ Step 01: You need to create a form

```
<form action="validate-success.html" method="post" onsubmit="return validateForm()">
    <table>
        <tr>
            <td align="right">Username:</td>
            <td>
                <input id="txtUsername" type="text" placeholder="Please input username">
            </td>
        </tr>
        <tr>
            <td align="right">Password:</td>
            <td>
                <input id="txtPassword" type="password" placeholder="Please input password">
            </td>
        </tr>
        <tr>
            <td colspan="2" align="center">
                <input type="submit" value="Register">
            </td>
        </tr>
    </table>
</form>
```

+ Step 02: You need to create a script for validation

```
<script>
    function validateForm(){
        var userName = document.getElementById("txtUsername").value;
        var passWord = document.getElementById("txtPassword").value;
        if (userName==null || userName==""){
            alert("Name can't be blank");
            return false;
        }else if(passWord.length < 6){
            alert("Password must be at least 6 characters long.");
            return false;
        }
    }
</script>
```

+ Step 03: You need to create an html page for validation result

```
    <style>
        div {
            border: 2px solid green;
            padding: 20px;
            background-color: aquamarine;
        }
        h1 {
            color: blue;
            text-align: center;
        }
        p {
            font-size: large;
            color: brown;
            font-style: italic;
            text-align: center;
        }
    </style>
</head>
<body>
    <div>
        <h1>Register successfully</h1>
        <p>Thank you so much</p>
    </div>
</body>
```

- The second way: You can validate form by tag. It will show result on html page (below input)

Instead of alert dialog, you will create div tag below input tag for validation. You can follow these steps: + Step 01: Create a form

```html
<form action="validate-success.html" method="post" onsubmit="return validateForm()">
    <table>
        <tr>
            <td align="right">Username:</td>
            <td>
                <input id="txtUsername" type="text" placeholder="Please input username" onblur="checkUsername()">
                <div id="messageUsername" class="error"></div>
            </td>
        </tr>
        <tr>
            <td align="right">Password:</td>
            <td>
                <input id="txtPassword" type="password" placeholder="Please input password" onblur="checkPassword()">
                <div id="messagePassword" class="error"></div>
            </td>
        </tr>
        <tr>
            <td colspan="2" align="center">
                <input type="submit" value="Register">
            </td>
        </tr>
    </table>
</form>
```

+ Step 02: You need to create a script for validation

```html
<style>
    .error {
        color: red;
    }
</style>
<script>
    function checkUsername(){
        var userName = document.getElementById("txtUsername").value;
        if (userName==null || userName==""){
            document.getElementById("messageUsername").innerHTML = "Name can't be blank";
            return false;
        }else{
            document.getElementById("messageUsername").innerHTML = "";
            return true;
        }
    }
    function checkPassword(){
        var passWord = document.getElementById("txtPassword").value;
        if (passWord.length < 6){
            document.getElementById("messagePassword").innerHTML = "Password must be at least 6 characters long.";
            return false;
        }else{
            document.getElementById("messagePassword").innerHTML = "";
            return true;
        }
    }
    function validateForm(){
        var isValidUsername = checkUsername();
        var isValidPassword = checkPassword();
        var isValid = isValidUsername && isValidPassword;
        return isValid;
    }
</script>
```

+ Step 03: Create an html page for validation result (same as above)

II/. Practice Javascript I

have a form like this:

I want to validate this form like this:

- User must enter Username before submit form. If username is not entered, an alert will show “Username can't be empty”.
- User must enter Password according to the rules: Password length must be from 6 to 20 characters. If user enter password wrong, an alert will show “Password length must be from 6 to 20 characters”.
- Please check confirm password and password are matched.
- Validate email:

There are many criteria that need to be follow to validate the email id such as:

- Email id must contain the @ and . character
- There must be at least one character before and after the @.
- There must be at least two characters after . (dot)

+ Valid emails: abc@abc.com

+ Invalid emails: 123@abc.com, @abc.com, abc@abc,....

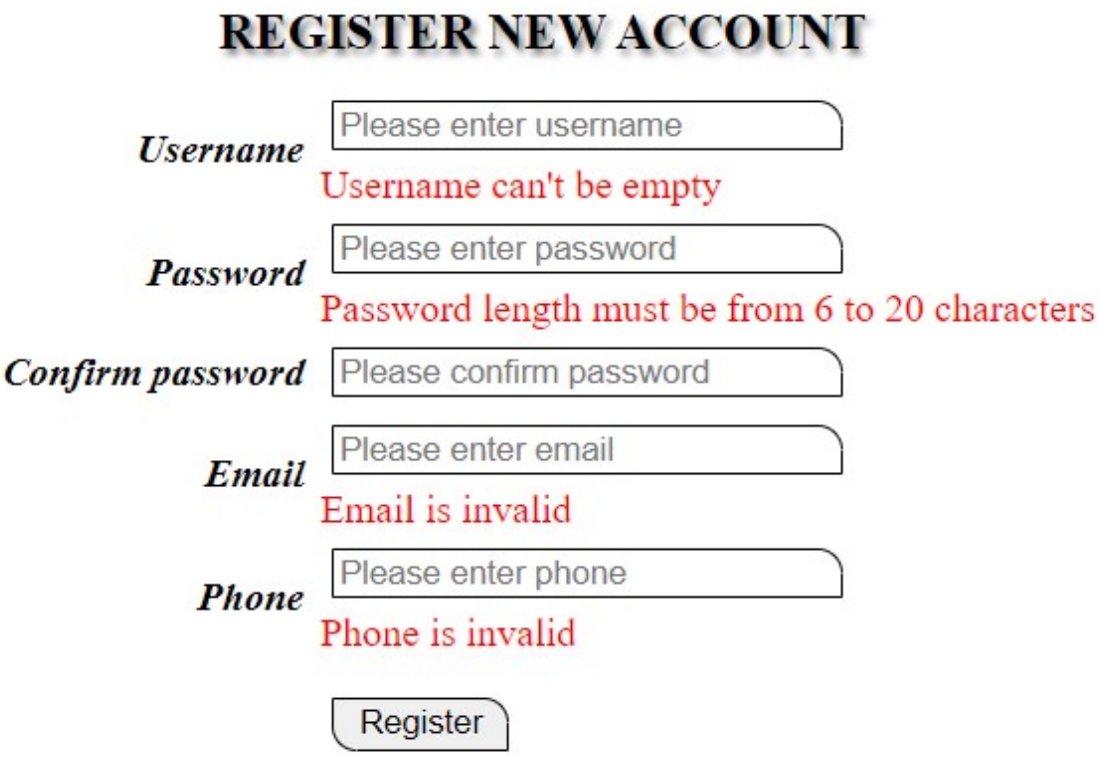
- Validate phone:

+ Start number must be 0

+ After 0, we must have at least 9 numbers and at most 10 numbers

When I click to submit form, I want you to validate form by 2 ways like this:

- The first way:



- The second way:

You can try:

- The first way:

+ Step 01: Create a form

```
<form action="register-process.html" method="post" onsubmit="return checkAllData()" autocomplete="on">
    <table border="0" cellpadding="0" cellspacing="2" align="center">
        <tr>
            <th colspan="2" class="header">Register new account</th>
        </tr>
        <tr>
            <td class="label">
                <label for="txtUsername">Username</label>
            </td>
            <td>
                <input type="text" name="txtUsername" id="txtUsername" placeholder="Please enter username" class="input" onblur="checkUsername()" />
                <div class="error" id="txtUsernameMessage"></div>
            </td>
        </tr>
        <tr>
            <td class="label">
                <label for="txtPassword1">Password</label>
            </td>
            <td>
                <input type="password" name="txtPassword1" id="txtPassword1" placeholder="Please enter password" class="input" onblur="checkPassword1()" />
                <div class="error" id="txtPassword1Message"></div>
            </td>
        </tr>
        <tr>
            <td class="label">
                <label for="txtPassword2">Confirm password</label>
            </td>
            <td>
                <input type="password" name="txtPassword2" id="txtPassword2" placeholder="Please confirm password" class="input" onblur="checkPassword2()" />
                <div class="error" id="txtPassword2Message"></div>
            </td>
        </tr>
```

```
        <tr>
            <td class="label">
                <label for="txtEmail">Email</label>
            </td>
            <td>
                <input type="text" name="txtEmail" id="txtEmail" placeholder="Please enter email" class="input" onblur="checkEmail()" />
                <div class="error" id="txtEmailMessage"></div>
            </td>
        </tr>
        <tr>
            <td class="label">
                <label for="txtPhone">Phone</label>
            </td>
            <td>
                <input type="text" name="txtPhone" id="txtPhone" placeholder="Please enter phone" class="input" onblur="checkPhone()" />
                <div class="error" id="txtPhoneMessage"></div>
            </td>
        </tr>
        <tr>
            <td></td>
            <td class="footer">
                <input type="submit" name="btnRegister" id="btnRegister" class="button" value="Register" />
            </td>
        </tr>
        <tr>
            <td class="error" colspan="2" id="txtError"></td>
        </tr>
    </table>
</form>
```

+ Step 02: Style a form

```
<style>
    .header {
        padding-top: 10px;
        padding-bottom: 10px;
        font-size: 20px;
        font-weight: bold;
        text-transform: uppercase;
        text-shadow: 2px 2px 3px #777;
        text-align: center;
    }

    .footer {
        padding-top: 10px;
        padding-bottom: 10px;
    }

    .label {
        text-align: right;
        padding: 5px;
        font-weight: bold;
        font-style: italic;
    }

    .input {
        margin: 5px;
        border-radius: 0px 10px 0px 0px;
        border: 1px solid #000;
        width: 200px;
    }

    .button {
        border: 1px solid #000;
        border-radius: 0px 10px 0px 10px;
        padding: 2px 10px;
        margin: 5px;
    }

    div.error {
        color: red;
    }
</style>
```

+ Step 03: Validate form

```
<script type="text/javascript">
    var patt_email = /^[a-zA-Z]+\w*([.-_]\w+)*\@[a-zA-Z]+\w*([,-_]\w+)*(\.\w+)+$/;
    var patt_phone = /^0[1-9]\d{8,10}$/;
    var errorMessage;

    function checkUsername() {
        tk = document.getElementById("txtUsername").value.trim();
        document.getElementById("txtUsernameMessage").innerHTML =
            tk == "" ? "Username can't be empty" : "";
        return (tk != "") ? true : false;
    }

    function checkPassword1() {
        mk1 = document.getElementById("txtPassword1").value;
        document.getElementById("txtPassword1Message").innerHTML =
            (mk1.length < 6 || mk1.length > 20) ?
            "Password length must be from 6 to 20 characters" : "";
        return (mk1.length >= 6 && mk1.length <= 20);
    }

    function checkPassword2() {
        mk1 = document.getElementById("txtPassword1").value;
        mk2 = document.getElementById("txtPassword2").value;
        document.getElementById("txtPassword2Message").innerHTML =
            (mk1 != mk2) ? "Password doesn't match" : "";
        return (mk1 == mk2);
    }

    function checkEmail() {
        email = document.getElementById("txtEmail").value;
        document.getElementById("txtEmailMessage").innerHTML =
            patt_email.test(email) == false ? "Email is invalid" : "";
        return patt_email.test(email);
    }
```

```
    function checkPhone() {
        phone = document.getElementById("txtPhone").value;
        document.getElementById("txtPhoneMessage").innerHTML =
            patt_phone.test(phone) == false ? "Phone is invalid" : "";
        return patt_phone.test(phone);
    }

    function checkAllData() {
        isValidUsername = checkUsername();
        isValidPassword1 = checkPassword1();
        isValidPassword2 = checkPassword2();
        isValidEmail = checkEmail();
        isValidPhone = checkPhone();

        isValid = isValidUsername &&
            isValidPassword1 &&
            isValidPassword2 &&
            isValidEmail &&
            isValidPhone;
        errorMessage = document.getElementById("txtError");
        if (isValid == true) {
            errorMessage.innerHTML = "Succesful!";
            errorMessage.style.color = "#0f0";
        } else {
            errorMessage.innerHTML = "";
        }
        return false;
    }
</script>
```

- The second way:

+ Step 01: Create a form

```
<form action="register-process.html" method="post" onsubmit="return checkAllData()">
    <table border="0" cellpadding="0" cellspacing="2" align="center">
        <tr>
            <th colspan="2" class="header">register new account</th>
        </tr>
        <tr>
            <td class="label">
                <label for="txtUsername">Username</label>
            </td>
            <td>
                <input type="text" name="txtUsername" id="txtUsername" placeholder="Please enter username" class="input" onblur="checkUsername()" />
            </td>
        </tr>
        <tr>
            <td class="label">
                <label for="txtPassword1">Password</label>
            </td>
            <td>
                <input type="password" name="txtPassword1" id="txtPassword1" placeholder="Please enter password" class="input" onblur="checkPassword1()" />
            </td>
        </tr>
        <tr>
            <td class="label">
                <label for="txtPassword2">Confirm password</label>
            </td>
            <td>
                <input type="password" name="txtPassword2" id="txtPassword2" placeholder="Please confirm password" class="input" onblur="checkPassword2()" />
            </td>
        </tr>
```

```
        <tr>
            <td class="label">
                <label for="txtEmail">Email</label>
            </td>
            <td>
                <input type="text" name="txtEmail" id="txtEmail" placeholder="Please enter email" class="input" onblur="checkEmail()" />
            </td>
        </tr>
        <tr>
            <td class="label">
                <label for="txtPhone">Phone</label>
            </td>
            <td>
                <input type="text" name="txtPhone" id="txtPhone" placeholder="Please enter phone" class="input" onblur="checkPhone()" />
            </td>
        </tr>
        <tr>
            <td></td>
            <td class="footer">
                <input type="submit" name="btnRegister" id="btnRegister" class="button" />
            </td>
        </tr>
        <tr>
            <td class="error" colspan="2" id="txtError"></td>
        </tr>
    </table>
</form>
```

+ Step 02: Style a form

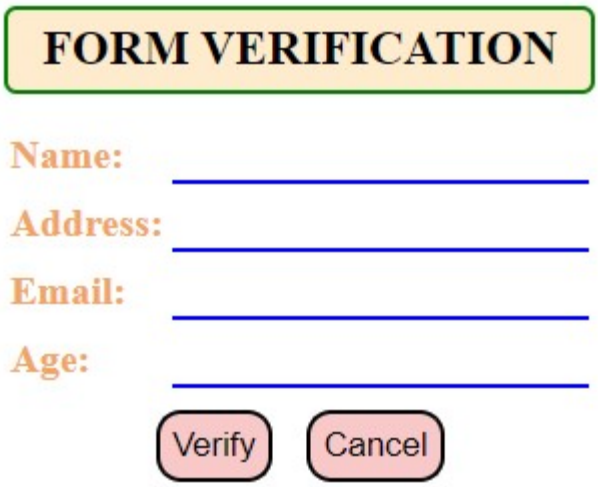
```
<style>
    .header {
        padding-top: 10px;
        padding-bottom: 10px;
        font-size: 20px;
        font-weight: bold;
        text-transform: uppercase;
        text-shadow: 2px 2px 3px #777;
        text-align: center;
    }

    .footer {
        padding-top: 10px;
        padding-bottom: 10px;
    }

    .label {
        text-align: right;
        padding: 5px;
        font-weight: bold;
        font-style: italic;
    }

    .input {
        margin: 5px;
        border-radius: 0px 10px 0px 0px;
        border: 1px solid #000;
        width: 200px;
    }

    .button {
        border: 1px solid #000;
        border-radius: 0px 10px 0px 10px;
        padding: 2px 10px;
        margin: 5px;
    }

    .error {
        color: red;
    }
</style>
```

+ Step 03: Validate form

```
<script type="text/javascript">
    var patt_email = /^[a-zA-Z]+\w*([.-_]\w+)*\@[a-zA-Z]+\w*([.-_]\w+)+$/;
    var patt_phone = /^0[1-9]\d{8,10}$/;

    var errorArray = [];

    function checkAllData() {
        tk = document.getElementById("txtUsername").value;
        mk1 = document.getElementById("txtPassword1").value;
        mk2 = document.getElementById("txtPassword2").value;
        email = document.getElementById("txtEmail").value;
        phone = document.getElementById("txtPhone").value;

        errorArray = [];
        if (tk.trim() == "")
            errorArray.push("Username can't be empty");
        if (mk1.length < 6 || mk1.length > 20)
            errorArray.push("Password length must be from 6 to 20 characters");
        if (mk1 != mk2)
            errorArray.push("Password doesn't match");
        if (patt_email.test(email) == false)
            errorArray.push("Email is invalid");
        if (patt_phone.test(phone) == false)
            errorArray.push("Phone number is invalid");

        errorMessage = document.getElementById("txtError");
        if (errorArray.length == 0) {
            errorMessage.innerHTML = "Succesful!";
            errorMessage.style.color = "#0f0";
            //return true;
            return false;
        } else {
            var strMsg = "Please check invalid data: ";
            strMsg += "<ol>";
            for (const index in errorArray) {
                strMsg += "<li>" + errorArray[index] + "</li>";
            }
            strMsg += "<ol>";
            errorMessage.innerHTML = strMsg;
            errorMessage.style.color = "#f00";
            return false;
        }
    }
</script>
```

III/. More Exercises

1/. Create a form like this:



Check input data:

- When clicking the Verify button, a message will be displayed telling the user whether they have filled in the correct information or not. The notice is as follows:

+ The name field cannot be left blank.

+ The address field cannot be left blank.

+ Email field cannot be left blank.

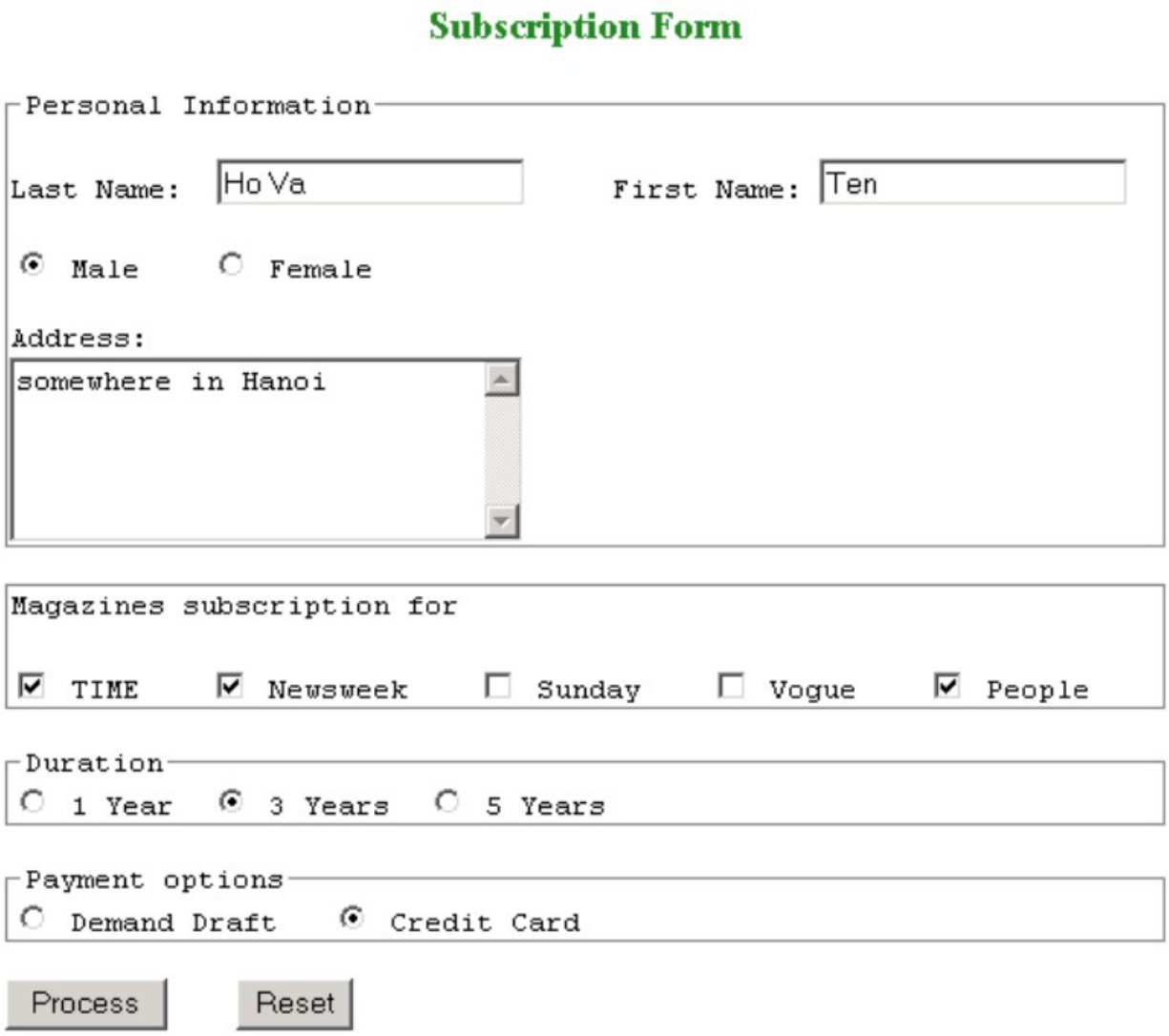
+ The email field must be in the correct email format.

+ The age field cannot be left blank.

+ The age field can only enter integers.

+ If the entered data is complete and correct, the message: Valid data will be displayed.

- When the Cancel button is pressed, all fields must be blank 2/. Create a form like this:



Check input data:

- When pressing the Process button if the form has not been filled in completely, a message will be displayed to the user.

+ The first name field cannot be left blank.

+ The last name field cannot be left blank.

+ The address field cannot be left blank.

+ User must select at least one journal.

- If the form is completely filled out, a confirmation dialog is displayed. Magazines (magazines) and payment method (payment) must display correctly.

+ If the OK button is then pressed, another message box must be displayed. Gender (gender) and address must display correctly.

+ If the Cancel button is then clicked, the last name field will receive focus and the user can change their selection.

- When the Reset button is pressed, all fields must be cleared.