

1. شرح محتوى الملف ValidatePassword.h

نبدأ مع تضمين المكتبات :

```
#include<conio.h>
#include<stdio.h>
#include<ctype.h>
#include<windows.h>
```

قمث بتضمين تلك المكتبات لأنني سأستخدم `getch & putchar` من `conio.h` و `printf & scanf` من `stdio.h` و `isdigit & iscntrl` من `ctype.h` و كل من الدالة `SetConsoleTextAttribute` و صديقتها `GetStdHandle` من `windows.h`.

```
#define MAX_PASSWORD_LENGTH 21
```

الثابت `MAX_PASSWORD_LENGTH` عبارة عن الطول الأقصى الذي يمكن أن تصل إليه كلمة المرور.

```
void colorString(int couleurDuTexte, const char* str = NULL)
```

الدالة `colorString` تستقبل رقم اللون و الجملة المراد تلوينها، ثم تقوم بإظهار الجملة باللون المطلوب. الوسيط الثاني للدالة هو وسيط افتراضي لذا يمكننا استدعاء الدالة بوسيط واحد وهو رقم اللون. يتم إظهار محتوى الوسيط الثاني إذا و فقط إذا كان يُمثل سلسلة محارف فعلية.

```
void printErrorPassword(int numError, char charError = '\0')
```

الدالة `printErrorPassword` مسئولة عن إظهار رسائل الخطأ المتعلقة بالإدخال. تستقبل الدالة وسيطين، أحدهما حقيقي و الآخر افتراضي، الوسيط الأول عبارة عن رقم الخطأ أما الثاني فيمثل المحرف المخالف لقواعد الإدخال. من الطبيعي جداً أن يكون الوسيط الثاني افتراضي لأنه يمثل حالة واحدة فقط (يتعلق برسالة خطأ واحدة) أما في بقية الحالات فسيتم استدعاء الدالة بوسيط واحد وهو رقم الخطأ.

في جسم الدالة نجد القرار `switch` الذي يأتي دوره في اختبار رقم الخطأ و إعطاء التحكم لل `case` المناسبة من أجل إظهار الرسالة التي تُوضح ماهية الخطأ.

و هذه ترجمة سريعة لرسائل الخطأ :

- ✓ **ال case الأولى :** لقد أدخلتم محرف غير مرئي.
- ✓ **ال case الثانية :** عفوا، الرمز المدخل غير صحيح. (تُخالف لقواعد كلمة المرور)
- ✓ **ال case الثالثة :** كلمة المرور المدخلة عبارة عن مسافات بيضاء فقط !
- ✓ **ال case الافتراضية (default) :** رقم الخطأ غير معروف.

الـ **case** الأخيرة تأتي فائدتها في مساعدة المبرمج إذا تم استدعاء الدالة برقم خطأ غير معروف.

```
void CheckLengthPassword(int numError)
```

الدالة **CheckLengthPassword** مسئولة عن إظهار رسائل الخطأ المتعلقة بطول كلمة المرور. تستقبل الدالة وسيط واحد وهو رقم الخطأ. الخطأ رقم 1 يعني أن طول كلمة المرور المدخلة أقل من 10 و الخطأ الثاني يعني أن طول الأخيرة أكبر من 30.

فكرة الـ **case** الأخيرة ذكرناها آنفا.

```
bool CheckMaxLength(int Length, int lengthSup)
```

الدالة **CheckMaxLength** تستقبل وسيطين. الأول عبارة عن طول كلمة المرور و الثاني يمثل الطول الأقصى. إذا كان الطول الحالي أكبر من الطول الأقصى ستعيد الدالة **true** و إلا فستعيد **false**.

```
bool CheckMinLength(int Length, int lengthInf)
```

نفس فكرة الدالة السابقة مع استبدال الطول الأقصى بالطول الأدنى.

```
bool HasNoneOf(char c, const char* str)
```

هذه الدالة تستقبل وسيطين، الأول عبارة عن محرف و الثاني سلسلة محارف، إذا كان الوسيط الأول ينتمي إلى الوسيط الثاني ستعيد الدالة **true** و إلا فستعيد **false**.

```
bool ValidatePassword(char* password)
```

هذه هي الدالة الرئيسية التي تتحقق من صحة كلمة المرور.

في البداية تم الإعلان عن سلسلة حرفية باسم **not_accepted_chars** و التي تحتوي على مجموعة الرموز المخالفة لقواعد كلمة المرور.

```
do {  
    ...  
} while (int(*password++) != 13);
```

يتم الخروج من الحلقة **do while** عندما يضغط المستخدم على **enter**.

إليك أخي القارئ شرح بسيط و مختصر للأوامر الموجودة داخل الحلقة :

المتغير **passwordLength** يمثل الطول الحالي لكلمة المرور بينما يتم تخزين عدد المسافات البيضاء التي تحتوي عليها كلمة المرور الحالية داخل المتغير **CheckSpace**.

تكون قيمة المتغير المنطقي `IsCntrl` مُساوية لـ `true` إذا وفقط إذا كان الرمز المُدخل عبارة عن أحد الرموز الغير مرئية. أما المتغير `CheckPasswordError` فتكون قيمته `true` إذا كان الرمز المُدخل غير مرئي أو ينتمي إلى الأحرف المحظورة في كلمة المرور.

في كل مرة، تتم قراءة الحرف الذي أدخله المستخدم ثم يُخزن هذا الحرف داخل سلسلة المحارف `password` ، بعد ذلك تعرض الدالة `ValidatePassword` على الشاشة إشارة نجمة. القراءة هنا تتم بواسطة الدالة `getch` و بالتالي لن يظهر الحرف المُدخل.

هذا السطر :

```
if(*password != ' ' & (int) *password != 13 & CheckPasswordError)
```

يعني : إذا كان الرمز المُدخل عبارة عن أحد الحروف المحظورة أو يمثل حرف غير مرئي باستثناء المسافة البيضاء و الـ `.enter`

هذا الشرط :

```
if(!IsCntrl)
```

يعني أن الرمز المُدخل ليس رمز غير مرئي و بالتالي فهو أحد الحروف المحظورة لذا قمنا باستدعاء الدالة `printErrorPassword` بوسيطين، الوسيط الأول هو الخطأ رقم 2 أما الوسيط الثاني فهو الرمز المُخالف.

`else` تعني أن الرمز المُدخل عبارة عن محرف غير مرئي و في هذه الحالة يتم استدعاء الدالة `printErrorPassword` بالخطأ رقم واحد.

بعد ذلك، سيتم تخزين الرمز `\0` الذي يدل على نهاية السلسلة الحرفية داخل الخانة الحالية بدلا من الحرف المُخالف، ثم تعيد الدالة `false`.

```
if (CheckMaxLength(passwordLength, 30)) {  
    CheckLengthPassword(2) ;  
    return false;  
}
```

في كل مرة أيضا سيتم فحص طول كلمة المرور، إذا تجاوز عدد حروفها 30 سيتم استدعاء الدالة `CheckLengthPassword` و إظهار الرسالة المتعلقة بالخطأ رقم 2 ثم تعيد الدالة `false`.

نأتي الآن لما بعد الحلقة `do while` :

الخروج من الحلقة يعني أن المستخدم ضغط على `enter` كما يعني أيضا أن طول كلمة المرور المدخلة أقل من الطول الأقصى. بقيت الآن حالتين للتحقق :

➤ هل طول كلمة المرور أقل من الطول الأدنى ؟

➤ هل كلمة المرور المدخلة تحتوي على مسافات بيضاء فقط ؟

قبل أن نبدأ التحقق يجب أن نضع الرمز `\0` في نهاية السلسلة، و هذا ما فعلناه في هذا السطر :

```
*(password - 1) = '\0';
```

السبب في كتابة `password - 1` بدلا من `password` بسيط جدا، وهو أنه بالخروج من الحلقة `do while` ستم إزاحة العنوان إلى الذي يليه، و بالتالي سيُشير المؤشر `password` إلى الخانة الموجودة بعد الخانة التي تحتوي على الكود رقم 13 (الذي يُقابله رمز الـ `enter` في الـ `ASCII`) .

هذا الشرط :

```
if(CheckSpace & CheckSpace == passwordLength - 1)
```

يعني : إذا كان المتغير `CheckSpace` لا يساوي صفر (أو لنقل غير معدوم و هذا معناه أن كلمة المرور غير خاوية) و عدد المسافات البيضاء يُساوي عدد خانات كلمة المرور (هذا معناه أن كلمة المرور عبارة عن مسافات بيضاء فقط) فقم باستدعاء الدالة `printErrorPassword` برقم الخطأ 3. في هذه الحالة ستعيد الدالة `.false`.

هذا الشرط :

```
if (CheckMinLength(passwordLength, 10))
```

يعني : إذا كان طول كلمة المرور المدخلة أقل من الطول الأدنى فقم باستدعاء الدالة `CheckLengthPassword` برقم الخطأ 1. في هذه الحالة ستعيد الدالة `.false`.

لم تبق الآن سوى حالة واحدة :

```
else return true
```

و تعني : كلمة المرور المدخلة صحيحة و توافق كافة الشروط، لذا ستعيد الدالة `.true`.