مبادئ الادخال والاخراج – Simple Input and Output

من أهم الأمور في البرامج الحقيقة هي أن يستطيع البرنامج التعامل مع الوسط المحيط به . في هذا الدرس سوف أقدم قسم صغير من مكتبة سي++ القياسية في الادخال والاخراج . لن أقوم بشرح كل التعليمات والعمليات الموجودة في هذه المكتبة وسوف أترك هذا إلى دروس لاحقة . الآن سوف أقوم بشرح العمليات اللازمة للتعامل مع لوحة المفاتيح والشاشة فقط . هذا سوف يعطينا خطوة أولية في بناء برمجيات تتعامل مع الأجهزة الطرفية السابقة. في دروس قادمة باذن الله سوف نكمل عمليات الادخال والاخراج حتى تضم التعامل مع الملفات على القرص أو العمليات على السلاسل .

3.1 الوصول إلى مكتبة سي++ القياسية – Access to the standard C++ library

في أي برنامج سي++ يتم تنفيذه فإن أربع متحولات يتم حجزها هذه المتحولات هي عبارة عن مسارات (streams) للمعلومات من وإلى الأجهزة الطرفية . اثنان من هذه المسارات يسميان ب standard input و standard output . لماذا سمينا المتحولات السابقة مسارات هذا بسبب أنها تمثل كمسار للبيانات بين البرنامج ومصدر البيانات أو وجهتها . المصدر أو الوجهة قد تكونا أحد الملفات أو حتى الذاكرة . ويمكن أيضا تحويل هذا المسار في أي وقت كان من مصدر إلى آخر أو من وجهة إلى أخرى . الافتراضي في السي++ أن يكونا المسارين الافتراضيين هما من الكيبورد (standard input) أو إلى الشاشة (standard output) . ولكن على حسب ضوء البرنامج فيمكن تحويل هذه المسارات لأن يكون الدخل الافتراضي من ملف معين ؛ وهذا ما نلاحظه إذا فكرنا في برمجة مترجم فالبديهي أن يكون مسار الدخل الافتراضي من الملف بدلا من الكيبورد .

جميع ملفات برامجك التي تستخدم مكتبة الدخل والخرج يجب أن تحتوي على هذا السطر في بداية الملف :

```
#include <iostream.h>
```

سوف نقوم بشرح عبارة #include في الدرس القادم . الآن أريدك أن تتقبل أن ملف التعريف iostream.h (header file) يحتوي على معلومات وتعريفات ضرورية من أجل استخدام مكتبة الدخل والخرج وهو الذي يقوم بتعريف المسارات التي تحدثنا عنها سابقا . هذا المعلومات جميعها سوف تضمن في الملف الحالي لأننا كتبنا عبارة #include في أعلى الملف .

3.2 الاخراج غير المنسق – Unformatted output

عمليات الاخراج تتم عن طريق المعامل >> وهي تدعى بمعامل الاضافة أو insertion operator . اسمه معامل الاضافة لأنه يستخدم في اضافة البيانات إلى ال stream أو المسار الذي يؤدي في النهاية إلى ظهور البيانات على الشاشة بشكل افتراضي . باستخدام المتحول cout الذي هو (standard output) المعرف في iostream.h و المعامل السابق يمكن تحويل البيانات إلى الشاشة بشكل افتراضي أو إلى أي stream آخر ، كما في المثال التالي :

```
cout << x;
```

إن معامل الاضافة تماما مثل بقية معاملات السي++ له أولوية وتكون دنيا ويتم تنفيذ أو تجميع التعبير من اليسار إلى اليمين. وبالتالي بما أن أولويته دنيا فإن العبارة التالية سوف تنفذ المطلوب باخراج ناتج جمع المتحولين x و y :

```
cout << x + y;
```

إن معامل الاضافة >> يمكن استخدامه أكثر من مرة تماما كما في سلسلة الاسنادات لأن هذا المعامل يرجع قيمة تشير إلى مسار الخرج المستخدم . وبما أن تجميع أو طريقة تنفيذ التعبير تتم من اليسار إلى اليمين فيمكن استخدامه في عملية اخراج أنواع مختلفة من البيانات إلى الشاشة كالتالي :

```
cout << "The sum of " << x << " and " << y << " is " << x + y;
```

أولا يتم اخراج الجملة "The sum of " إلى ال stream ونتيجة المعامل هي عبارة عن مؤشر إلى ال stream المستخدم هنا وهو cout . هذه الأخيرة سوف تستخدم مرة أخرى من أجل اخراج المتحول x وهكذا ..

3.3 الادخال غير المنسق – Unformatted input

عمليات الادخال تتم عن طريق المعامل >> وهي تدعى بمعامل الاستخراج أو extraction operator . اسمه معامل الاستخراج لأنه يستخدم في استخراج البيانات من ال stream أو المسار إلى متحول في البرنامج . باستخدام المتحول cin الذي هو (standard input) المعرف مسبقا في iostream.h جنبا إلى جنب مع المعامل السابق يمكن استخلاص البيانات من لوحة المفاتيح بشكل افتراضي أو من أي stream آخر ، كما في المثال التالي :

```
cin >> x;
```

إن معامل الاستخراج أولويته تماما مثل معامل الاضافة وأيضا بالنسبة إلى كيفية تنفيذه أو تجميعه . أيضا هذا المعامل يرجع قيمة تشير إلى مسار الدخل المستخدم وبالتالي يمكن لهذا المعامل أن يتم استخدامه كسلسلة من الادخالات كالتالي :

```
cout << "Please enter three numbers : ";
cin >> first >> second >> third;
```

إن هذا المعامل متوافق مع جميع الأنواع الأساسية في سي++ . يقوم هذا المعامل باهمال أي مسافات أو محارف ال (tab) أو محارف (new line) حتى يتم قرائة المحارف المتوافقة مع النوع الذي نريد قرائته . على سبيل المثال إذا كانت first و second هما من نوع int أما third فهي من نوع سلسلة محارف . وكان الدخل على الصورة التالية :

```
167     34hello
```

فإن سوف يتم استخراج القيمة 167 ووضعها في first وبعدها يتم اهمال المسافات التي بعدها حتى يتم قرائة القيمة 34 ووضعها في المتحول second . لاحظ أنه لم يتم قرائة hello مع أنها موصولة مع القيمة 34 لأنها من نوع آخر . وأخيرا يتم قرائة hello ووضعها في المتحول third .

3.4 تنسيق الخرج – Manipulating the output

إن مكتبة الدخل والخرج تحتوي على عدة طرق من أجل تنسيق الخرج إلى ال stream من داخل البرنامج . سوف نشرح هنا كيفية استخدام المعدلات أو Manipulators . سوف أكمل شرح هذه المعدلات وطرق أخرى في دروس قادمة باذن الله . أما هنا فسوف أقدم طريقة سهلة لتنسيق الخرج . وهذا المنسق يظهر في عبارة الخرج تماما مثل بقية الحدود في الخرج ولكنه لا يخرج أي شئ إلى الstream بل له تأثير على الكمية التي تخرج بعده . مثلا المثال التالي :

```
int index = 10;
cout << "[" << index << "]";
```

سوف ينتج لنا الخرج التالي :

```
[10]
```

على فرض أننا نريد طباعة index في مساحة تكفي لجميع القيم الممكنة لها فإننا نستخدم التالي :

```
cout << "[" << setw(4) << index << "]";
```

سوف نحصل على :

```
[  10]
```

حيث تقوم setw باخبار أن تظهر القيمة التي بعدها في مساحة مكونة من أربع محارف . أما إن كانت القيمة index تحتاج إلى أكثر من ذلك فطبعا سوف يتم طباعة الرقم كامل .

بالنسبة إلى طباعة الأرقام الحقيقة فإنه بشكل افتراضي يتم طباعة ستة أرقام بعد الفاصلة ويمكن تغيير هذا باستخدام المعدل setprecision كالتالي :

```
double pi = 3.1415926;
cout << setprecision(4) << pi;
```

سوف تحدد أننا مهتمين فقط بأربع أرقام كمجموع كلي وبالتالي سوف نحصل على الخرج التالي :

```
3.142
```

لاحظ كيف أنه تم تقريب العدد قبل الاخراج . يجب أن ننتبه إلى أمر هام وهو من أجل استخدام هذه المعدلات أو غيرها يجب تضمين الملف iomanip.h في البرنامج :

```
#include <iostream.h>
#include <iomanip.h>
```

3.5 التابعين get و put – The functions get and put

أحد الميزات السهلة التي لاغنى عنها أحيانا وهي كيفية اخراج أو ادخال محرف وحيد . بالنسبة للاخراج فهذا يتم عن طريق التابع put وهذا مثال على هذا :

```
char  ch = 'h';
cout.put(ch);
```

إن التابع السابق put يأخذ متحول واحد ويطبعه في cout أو standard output stream . إن كيفية الكتابة السابقة سوف تتعرف عليها أكثر عندما نبدأ في دروس عن ال classes لاحقا .

التابع get يقرأ المحرف التالي من standard input stream ويضعه في متحول ما كالتالي :

```
char ch;
cin.get(ch);
```

3.6 استخدام مكتبة الدخل والخرج الجديدة – Using the new standard I/O stream library

لا يوجد لغة من اللغات تتطور أو يضاف لها توابع داعمة جديدة ، وكذلك المكتبة القياسية في الادخال والاخراج في السي++ . ومن أجل استخدام آخر النسخ من هذه المكتبة في برنامجك فيجب أن يتم التضمين بالطريقة التالية :

```
#include <iostream>
using namespace std;
```

لاحظ أننا لم نستخدم الامتداد h للملف iostream وهذا مايخبر المترجم أننا نريد استخدام النسخة الحديثة من المكتبة . أما السطر الثاني فهو ضروري وسوف نتقبله حاليا مثل ماهو حتى نأتي لشرحه لاحقا في دروس أخرى . إذا يجب أن تنتبه أنه إذا أردت أن تستخدم مكتبات السي++ الجديدة قم بتضمين ملفات التعريف بدون استخدام الامتداد h حتى يعرف المترجم أي من المكتبات سوف يربطها مع برنامجك . مثلا إذا أردت استخدام المعدلات في الخرج سوف تكتب التالي :

```
#include <iostream>
#include <iomanip>
using namespace std;
```

في الدروس القادمة سوف نستخدم هذه الصيغة في تضمين مكتبة الدخل والخرج .

3.6 ملخص الدرس

- ميزات الدخل والخرج في سي++ يتم الحصول عليها من مكتبة الدخل والخرج القياسية Standard I/O stream library .
- عند استخدام المكتبة السابقة يجب تضمين الملف iostream.h مع البرنامج باستخدام العبارة #include<iostream.h>
- يمكن تنسيق الخرج باستخدام المعدلات manipulators .
- يفضل استخدام التابعين get و put من أجل تبادل محرف وحيد مع المسار (stream) .