وزارة التعليم العالي والبحث العلمي
جامعة ديالى
كلية التربية للعلوم الصرفة
قسم علوم الحاسوب

ادارة مطعم الآكلات السريعة باستخدام المحاكاة

بحث مقدم

الى كلية التربية للعلوم الصرفة قسم علوم الحاسوب

وهو جزء من متطلبات نيل شهادة البكالوريوس

في تربية علوم الحاسوب

اعداد الطالبة

امنة عبد الكريم يحيى

بأشراف

م.مقداد قيس

٢٠١٩ م ١٤٤٠ ه

الفصل الثاني

توليد الأرقام العشوائية

ما هو العدد العشوائي ؟ وهل هو موجود ؟

عندما نقول "عدد عشوائي" فهذا لا يعني بالضرورة أنه عشوائي بالمعنى العام .. العدد العشوائي هو العدد المولد من طرف دالة مجهولة السلوك (نسبيا) و عندما أقول نسبيا فأنا أقصد "المستخدم" ، أي أن المستخدم عندما يقوم بتشغيل الدالة ستقوم بإرجاع عدد عشوائي كما يتصور "هو" و لكن في الحقيقة .. هذا العدد العشوائي يخضع لقانون معين ..المبرمج فقط هو من يعرفه ! . فالمبرمجون أرادوا برمجة دالة تمكن من توليد أرقام عشوائية ! وكانت المشكلة تكمن في برمجة الدالة العشوائية .. فنحن حين نقول "دالة مبرمجة" فهذا يعني أن الدالة تخضع لقانون محدد .. أما العدد العشوائي فلا يخضع لأي قانون .. وإلا فسيكون عددا محددا و سيفقد عشوائيته .. هنا تكمن المشكلة.

كيف يمكننا توليد أرقام عشوائية ؟

يمكن توليد الأرقام العشوائية بأكثر من (قانون ،خوارزمية، طريقة) و كلما كان القانون أكثر تعقيدا يكون العدد أقرب ما يكون للعشوائية .. يعني يصبح من المستحيل (بالنسبة للمستخدم) معرفة سلوك الدالة التي تتولى المهمة .. فنحن عندما نقوم بإنشاء دالة عشوائية ترجع قيمة العدد المُدخل قسمة على ٢.

الآن أصبح من السهل اكتشاف سلوك هذه الدالة .. فعندما يقوم المستخدم بتشغيل هذه الدالة ١٠ مرات (مثلا) سيتمكن من استنتاج سلوك هذه الدالة بكل سهولة .. هنا يكمن احتياجنا في قانون رياضي يكون معقد بعض الشيء ..مما يربك المستخدم أثناء محاولته معرفة سلوك الدالة. الكلام السابق يكون أكثر دقة إذا كان العدد العشوائي عبارة عن عدد "كسري" وكلما كانت الفاصلة غير منتهية كان الكلام أدق .. أما إذا كان العدد العشوائي عبارة عن عدد صحيح .. فسيكون الكلام السابق غير دقيق تماما ! لأن المستخدم أصبح بإمكانه اكتشاف سلوك الدالة التي تولد الأعداد العشوائية.

فيمكنه (مثلا) توليد ١٠٠٠ عدد بين ١ و ١٠ مما يقوده إلى استنتاج سلوك الدالة .. وهكذا .. يعني يختار مجال ضيق ويقوم بتوليد الكثير من الأعداد العشوائية في المجال المُختار .. مُستنتجا الملامح الأولى لسلوك هذه الدالة.

الآن .. سنقوم باستعمال دالة تقوم بتوليد رقم عشوائي.

الدالة التي سنستعملها هي الدالة rand وتوجد في المكتبة cstdlib ، هذه الدالة تتبع خوارزمية مُعقدة بعض الشيء، مما يجعل عملية تتبع الرقم المُولد من طرف الدالة "عملية صعبة" إن لم تكن "مستحيلة" ، هذا في حالة استخدام البذرة أما إذا لم نستخدمها فستقوم الدالة بتوليد أرقام عشوائية "شبه منتظمة" و السبب هو أنها تتبع خوارزمية ثابتة لتوليد الأرقام العشوائية مثلا .. إذا قام المستخدم بأخذ مجال ضيق للأعداد الصحيحة و قام بتوليد الكثير من الأرقام العشوائية ثم

قام بتكرار العملية عدة مرات .. فسيلاحظ انتظام هذه الأعداد بشكل أو بآخر .. لذلك قام المبرمجون بإنشاء الدالة srand التي تعتمد على توليد الأرقام العشوائية .

الفكرة هنا تكمن في توليد رقم عشوائي يقع ضمن المجال الذي يتحرك فيه دليل عناصر المصفوفة ثم نقوم بطباعة العنصر الموافق للرقم العشوائي المولد مثلا اذا كان الرقم المولد هو ٢ فسيضهر الحرف c لانه يأخذ الترتيب ٢ في المصفوفة.

توليد الأرقام العشوائية كمومياً

توليد الأرقام العشوائية كمومياً مولد أرقام عشوائية كمومي يجمع أفضل ما في النهجين إن العلم نهج غالباً ما يسعى إلى النظام والأنماط الموجودة في العالم المحيط بنا، لكن للعشوائية (Randomness) استخداماتها أيضاً، فالأعداد العشوائية أداة مهمة جداً في مجالات مثل التشفير (Cryptography)، والنمذجة الحاسوبية (Computer Simulations)، والتحليل الإحصائي (Analysis Statistical)، ولعل توليد سلسلة محارف طويلة من أرقام عشوائية أمرٌ صعب ، وهو ضروري لتحقيق أداء أفضل والوصول إلى أمانٍ أكبر في تلك التطبيقات. تتضمن إحدى طرق توليد الأرقام العشوائية الاستفادة من العشوائية المتأصلة في أنظمة كمومية (Quantum System) معروفة بالضجيج الكمومي (Quantum Noise)، وتتضمن إجراءات التوليد الكمومي للأرقام العشوائية (QRNG) مصادر فوتونية مفردة، ولأنّ إصدار الفوتونات المفردة يجري عند أزمنة عشوائية، فمن المستحيل القيام بتحديد مثالي لعدد الفوتونات الصادرة عند زمنٍ محدد، ما يؤدي إلى وجود ارتياب في القياس وعشوائية أيضاً. تنتمي الطرق المستخدمة حالياً في مجال QRNG إلى نوعين: طرق تعتمد على الجهاز (Device-Dependent)، وطرق لا تعتمد على الجهاز (Device-Independent)، حيث تتطلب الطرق التي تعتمد على الجهاز، والمستخدمة في كل التطبيقات التجارية لـ QRNGs، وجود معرفة تفصيلية بآلية عمل الأجهزة المستخدمة في البروتوكول (Protocol)، وتُولد هذه الطريقة الأرقام العشوائية عند معدلات مرتفعة جداً (٤ مليون بت عشوائي في الثانية الواحدة) وعند مستوى أمان أقوى بكثير من ذلك الموجود في مولدات الأرقام التقليدية شبه العشوائية (Classical Pseudo-Random Number Generators)، لكنها تعتمد على افتراضات من الصعب التحقق منها. ومن ناحية أخرى، لا تتطلب الطرق التي لا تعتمد على الجهاز وجود نفس المعرفة بالأجهزة وهي تُقدم أماناً أقوى كذلك، إلا أن التطبيق العملي لها يتطلب وجود إعدادات متطورة ومعقدة جداً، ولا يُمكن إنجاز ذلك إلا عند معدلات منخفضة جداً لتوليد الأرقام العشوائية. وفي ورقة علمية جديدة نُشرت في مجلة "Physical Review Letters"، قام علماء فيزياء من جامعة جنيف بتطوير بروتوكول يُقدم نهجاً وسيطاً للوصول إلى QRNG، وهو يتطلب وجود افتراضات عامة قليلة حول الأجهزة، لكنه ليس بحاجة إلى نموذج تفصيلي خاص بعملها، ويقع كلٌ من معدل أداء هذا النهج (٢٣ بت عشوائي في الثانية الواحدة) ومستوى أمانه بين المستويات الخاصة بالطرق التي تعتمد على الجهاز وتلك المستقلة عنه؛ لكنّ هذا النهج يشترك مع الأولى بإمكانية تطبيقه بوجود التكنولوجيا القياسية الحالية. يقول نيكولا برونر (Nicolas Brunner)، المؤلف المشارك في الدراسة من جامعة جنيف لـ Phys.org: "قد تكون الأهمية الأساسية لعملنا هي القدرة على

التحقق من حصول عملية التوليد العشوائي للأرقام ضمن سيناريو تعاني فيه الأجهزة من عيوب تقنية، لكنها ليست ضارة بالنسبة للمستخدم". ويُضيف قائلاً: "إنه سيناريو يقع في مكانٍ ما بين الطرق المعتمدة على الأجهزة، التي يُفترض فيها أن الأجهزة موصوفة بشكلٍ جيد، وبين تلك المستقلة عن الأجهزة التي يُمكن فيها للعد من حيث المبدأ إدارة الجهاز". يكمن التحسين الأساسي في البروتوكول الجديد في كونه ذاتي الاختبار (Self-testing)، أي أن باستطاعته تقديم تقدير بالزمن الحقيقي لعشوائية بيانات الفوتونات التجريبية بالاعتماد على قياس الانتروبي (Entropy) الخاص بها، كما يُمكن للنهج الجديد التمييز بين هذه العشوائية الحقيقية وتلك الناجمة عن مصادر عشوائية أخرى مثل العيوب التقنية (Technical Imperfections). عندما تتم معرفة مقدار العشوائية الحقيقية، يُمكن بعدها معالجة البيانات الخام بشكلٍ مناسب لتوليد محارف من الأرقام العشوائية، ومن أجل إثبات القدرة الذاتية للاختبار، فقد أطفأ الباحثون وببساطة مكيف الهواء في الغرفة، وبسبب الحساسية الشديدة لأنظمة كمومية كتلك المعتمدة على مصادر فوتونات مفردة لبيئاتها، فإن التغير الحاصل في درجة الحرارة يؤثر على تحاذي التجهيز البصري عشوائية الفوتونات الصادرة، ومن ثم يستطيع النظام التعرف مباشرةً على أي تغير في عشوائية التوليد بحيث يُمكن تطبيق المزيد من المعالجة، وضمان استمرار الحصول على جيل من الأرقام العشوائية عالية النوعية. وبشكل عام، فإن البروتوكول الجديد يقدم طريقة QRNG مبسطة على الرغم من عدم تحقيقها لمعدلات مرتفعة كتلك الموجودة في التطبيقات التجارية لـ QRNGs، إلا أنها تؤدي إلى الحصول على مستوى أمني أقوى ودون الحاجة إلى التوصيف التفصيلي للأجهزة، وقد يكون هذا الجمع بين المميزات مفيداً جداً للتطبيقات المستقبلية. يقول برونر: "إن العشوائية مصدر مهم جداً للعديد من التطبيقات". ويتابع قائلاً: "ومع ذلك، فإنّ موثوقية العشوائية لا تزال تُمثل تحدياً مهماً؛ أي بمعنى آخر القدرة على تحديد مدى عشوائية مخرجات بعض الأجهزة بالاعتماد على افتراضات بسيطة يُمكن التحقق منها". ويختم برونر قائلاً: "إن هدفنا هو تطوير مخططات أفضل، يُمكن استخدامها عملياً بشكلٍ أسهل وتُحقق معدلات أكثر ارتفاعاً بكثير، ومع ذلك، فالهدف الرئيسي لا يزال يكمن في إيجاد سيناريو يُقدم الحل الأمثل عند المفاضلة بين الأمان وسهولة التطبيق".

الأعداد العشوائية وشبه العشوائية

تفيد الأرقام العشوائية في مجموعة متنوعة الأغراض والأهداف، مثل توليد مفاتيح لتشفير البيانات، والنمذجة ومحاكاة الظواهر المعقدة، واختيار عينات عشوائية من مجموعات البيانات الضخمة، وقد استُخدمَت من الناحية الجمالية في الأدب والموسيقا، وهي مشهورة الاستخدام في الألعاب والقمار أيضًا. فالرقم العشوائي هو أحد الأرقام التي يُحصَل عليها من مجموعة من القيم الممكنة، وكل رقم من المجموعة له احتمال متساوٍ مع البقية للحصول عليه، أي تتوزع احتمالاتها توزيعًا منتظمًا، فعند مناقشة سلسلة من الأرقام العشوائية يجب أن يكون كل عدد مُستخرَج مستقل إحصائيًّا عن الآخرين، أي إن الحصول على عدد ما لا يؤثر في احتمال الحصول على عدد آخر.

ومع ظهور الحواسيب، احتاج المبرمجون إلى وسيلة لاستحداث وتوليد العشوائية في برامج الحاسوب، وعلى رغم من ذلك، من الصعب إيجاد حاسوب يؤدي شيئًا مصادفة أو عشوائيًّا، لأن الحاسوب ينفذ التعليمات بصورة عمياء محددة له مسبقًا فيمكن التنبؤ بها تمامًا.

وهناك نوعان من الطرائق الرئيسية لتوليد الأرقام العشوائية باستخدام الحاسوب:

- مولدات أرقام شبه عشوائية (PRNGs – Pseudo Random Numbers Generators).

- مولدات أرقام عشوائية حقيقية (TRNGs – True Random Numbers Generators).

ولكل طريقة خصائص مختلفة تمامًا عن الأخرى ولكل منها إيجابياتها وسلبياتها، سنتحدث قليلًا عن النوعين:

توليد أرقام شبه عشوائية (PRNGs):

تدل كلمة "شبه" إلى أن الأرقام شبه العشوائية ليست عشوائية بالطريقة التي قد نتوقعها، على الأقل إذا اعتدنا عشوائية رمي حجر النرد أو بطاقات اليانصيب، فيكون أساسًا توليد أرقام شبه عشوائية عن طريق خوارزميات تستخدم صيغ رياضية أو تكون عبارة عن جداول محسوبة مسبقًا لإنتاج سلاسل من الأرقام التي تبدو عشوائية، أي بعبارة أخرى: تُولَّد الأعداد بطريقة محددة مسبقًا، وهي ليست عشوائية مطلقًا، فكل ما في الأمر أننا نحن البشر لا ندرك أن هناك علاقة رياضية تحكمها.

ومن الطرائق الجيدة لتوليد أرقام شبه عشوائية هي طريقة المطابقة الخطية التي تعتمد على معادلة خطية متقطعة، وهي تمثل واحدة من أقدم وأفضل الخوارزميات لتوليد مثل هذه الأعداد العشوائية، وقد أُجريَت بحوث كثيرة في نظرية توليد الأرقام شبه العشوائية، حتى أن الخوارزميات الحديثة لتوليد الأرقام شبه العشوائية الجيدة تكون الأرقام المولدة بها تبدو تمامًا أنها عشوائية (ولكنها ليست كذلك طبعًا).

والفرق الأساسي بين توليد أرقام شبه عشوائية وتوليد أرقام عشوائية حقيقية سهل الفهم إذا قارنَّا أرقام الحاسوب المولدة عشوائيًّا بالأرقام الناتجة عن رمي حجر النرد، ولأن توليد الأرقام شبه العشوائية يكون باستخدام المعادلات الرياضية أو القوائم المحسوبة مسبقًا، ونرمي حجر النرد عدة مرات وتسجيل النتائج وكلما رميته يمكنك الحصول على النتيجة التالية ضمن القائمة، ولكن قد لا يبدو الفرق واضحًا حتى الآن.

وتولد الأرقام شبه العشوائية بفعالية ولكنها محددة سلفًا، ولكن مولدات الأرقام العشوائية الحقيقة TRNGs تجعل الحاسوب يعمل بطريقة حجر النرد العشوائية تمامًا، وهي إحدى الطرائق الأكثر شيوعًا، أو يمكن استخدام بعض الظواهر الفيزيائية الأخرى التي يمكن إدخال نتائجها إلى الحاسوب بسهولة عن طريق بعض أجهزة القياس الفيزيائية، كدرجة الحرارة أو سرعة الرياح مثلًا (هذه الظواهر يمكن التنبؤ بها بطرق رياضية برمجية، ولكن لا يمكن تحديدها بدقة تامة أبدًا).

وتتسم PRNGs بالكفاءة، وهذا يعني أنها يمكن أن تنتج العديد من الأرقام في وقت قصير، وهي خاصية رائعة إذا كان التطبيق يحتاج إلى كثيرٍ من الأرقام، وتتسم بالحتمية أيضًا فأيُّ تسلسل معين من الأرقام يمكن أن يتكرر في وقت لاحق إذا كانت نقطة البداية في التسلسل معروفة، وتفيد هذه الخاصية إذا كنا بحاجة إلى تكرار نفس التسلسل من الأرقام مرة أخرى في مرحلة لاحقة ضمن التطبيق أو البرنامج.

وعادة ما تكون PRNGs دورية أيضًا، مما يعني أن التسلسل سوف يعيد نفسه في نهاية المطاف، وهي سمة نادرة مرغوب فيها.

وهذه الخصائص تجعل من PRNGs مناسبة للتطبيقات مثل المحاكاة ونمذجة التطبيقات إذ تتطلب كثير من الأرقام وتحتاج تكرار نفس التسلسل بعد فترة، وهي ليست مناسبة للتطبيقات التي من المهم أن تكون الأرقام غير متوقعة ولا يمكن التنبؤ بها، مثل تشفير البيانات والقمار.

وعلى الرغم من ذلك، يجب الإشارة إلى أنه رغم فعالية خوارزميات PRNGs الموجودة، لا تُستخدَم دائمًا؛ إذ إنها قد تعطينا نتائج سيئة، فمثلًا، برمجة مواقع باستخدام إحدى لغات البرمجة (وهي PHP): إذا كنا نستخدمها على منصة نظام لجنو/لينكس، هناك احتمال أن نكون موفقين بنتائج الأرقام العشوائية المولدة، أما إذا استخدمناها على منصة مايكروسوفت ويندوز، ربما سنجد أن الارقام العشوائية المولدة لم تصل إلى المستوى المطلوب كما هو موضح في التحليل البصري الذي أُجرِيَ عام ٢٠٠٨، وتظهر الصورة فرق التحليل البصري للغة PHP على منصة مايكروسوفت ويندوز مع موقع Random.org (وهو موقع يقدم خدمة توليد أعداد عشوائية حقيقة اعتمادًا على ظواهر فيزيائية)؛ إذ إن الطريقة الوحيدة لفحص فعالية توليد الأرقام العشوائية هو إنشاء صورة أو شكل مرئي للأرقام التي تنتجها، وذلك لأن البشر بارعون باكتشاف النماذج والروابط الشكلية في الصور والرسومات، ولكن لا يمكن عدُّ هذا النهج تحليلًا شاملًا بل هو تحليل غير رسمي، ولكنه يعطي طريقة سريعة للحصول على انطباع تقريبي لأداء المولِّدات.

والصورة التي على اليسار تظهر توليد أرقام عشوائية حقيقية من موقع Random.org، أما الصورة التي على اليمين تظهر تابع (rand) ضمن اللغة البرمجة PHP على منصة مايكروسوفت ويندوز الذي يُولِّد أرقامًا شبه عشوائية.

وتظهر الصورة الأرقام المولدة التي أنشأها PHP أنماطًا واضحة مقارنة مع الناتجة عن Random.org ذات مولد الأرقام العشوائية الحقيقية:

فمقارنة مع PRNGs، تستخرج TRNGs العشوائية من الظواهر الفيزيائية وتدخلها إلى جهاز الحاسوب؛ إذ يمكننا تخيل ذلك على أنه حجر نرد متصل بالحاسوب، ولكن فعليًّا يستخدم البشرُ الظواهرَ الفيزيائية التي يكون إدخالها إلى الحاسوب أسهل من حجر النرد، وقد تكون الظاهرة الفيزيائية بسيطة، مثل التغييرات الصغيرة لحركة الفأرة على شاشة حاسوبك أو مقدار الوقت بين ضغطات مفاتيح الحاسوب.

ومن الناحية العملية، علينا أن نكون حذرين بخصوص اختيار المصدر، فمثلًا، من الصعب استخدام ضغطات المفاتيح بهذه الطريقة، لأنه غالبًا ما يُخزِّنها نظام تشغيل الحاسوب مؤقتًا، مما يعني أن عدة ضغطات للمفاتيح تُجَمَّع قبل أن تُرسَل إلى البرنامج الذي يكون بحالة انتظار؛ إذ ينتظر البرنامج ضغطات المفاتيح، فيبدو وكأنه قد ضُغِط على المفاتيح في وقت واحد تقريبًا، وهذا يؤدي إلى عدم وجود كثيرٍ من العشوائية في هذه العملية.

ومع ذلك، هناك العديد من الطرق الأخرى للحصول على العشوائية حقيقية ضمن جهاز الحاسوب. إن المصدر المشع هو واحد من الظواهر الفيزيائية الجيدة للاستخدام؛ إذ مع مرور الزمن يتحلل المصدر الإشعاعي وهي عملية فيزيائية عشوائية، لا يمكن التنبوء بها ولكن يكشف عنها بسهولة تامة وتُستخدَم لتغذية الحاسوب بالنتائج.

وإنَّ خدمة HotBits في Fourmilab في سويسرا هي مثال ممتاز على توليد أرقام عشوائية باستخدام هذه التقنية.

وهناك ظاهرة فيزيائية مناسبة أخرى وهي ضجيج الغلاف الجوي، والتي من السهل جدًّا التقاطها من الراديو العادي، وهو النهج المتبع من قبل موقع Random.org. ويمكن استخدام الضوضاء الناتجة في المكتب أو المختبر أيضًا، ولكن سينتج عنه نوع من نمطية ما، ومن ثم تصبح PRNGs.

وتساهم مروحة جهاز الحاسوب في إنتاج الضوضاء، ونظرًا أنَّ المروحة جهاز دوري، هناك احتمال أن تكون الضوضاء التي تنتجها غير عشوائية كضوضاء الغلاف الجوي.

ومن أروع الطرق المستخدمة سابقًا، النهج البصري المولد عن Lavarand ، وهو عبارة عن جهاز عتادي لتوليد الأرقام العشوائية صممته شركة سيليكون غرافيكس، ويلتقط الصور للنماذج المقدمة من المواد العائمة لـ Lava Lamb وهو مصباح يحتوي على النقط الملونة من الشمع داخل مصباح زجاجي مملوء بسائل واضح وشفاف، ويرتفع الشمع ويسقط ضمن المصباح حسب التغييرات كثافته بسبب الحرارة ناتجة عن الضوء الساطع الواقع تحت المصباح، ويستخرَج البيانات العشوائية من الصور ويستخدمها كبذرة (أي بداية سلسلة) لتوليد أرقام عشوائية حقيقية.

وهناك نهج آخر هو Java Entropy Pool الذي يجمع بتات عشوائية من مجموعة متنوعة من المصادر، بما فيها موقعي HotBits و Random.org.

فبغض النظر عن أي ظاهرة فيزيائية تُستخدَم، فإن عملية توليد الأرقام العشوائية الحقيقية تنطوي على قليل من التحديد، وتغيرات غير متوقعة في البيانات، فعلى سبيل المثال، يستخدم موقع HotBits تغييرات صغيرة في التأخير الزمني أثناء حدوث التحلل الإشعاعي، ويستخدم موقع Random.org تغييرات صغيرة في سعة ضوضاء الغلاف الجوي.

تختلف خصائص TRNGs تمامًا عن PRNGs، فعادة ما تكون TRNGs غير فعالة مقارنة مع PRGNs، وتأخذ وقتًا طويلًا بكثير لإنتاج الأرقام وهي غير حتمية، أي إن تسلسل معين من الأرقام لا يمكن إعادة إنتاجه مرة أخرى، على الرغم من أن نفس التسلسل قد يحدث عدة مرات عن طريق الصدفة فقط، ثم إن TRNGs غير دورية.

المقارنة بين PRNGs و TRNGs:

يلخص الجدول أدناه ما وصلنا إليه من الخصائص لهذين النوعين من المولدات الأرقام العشوائية:

خصائص	مولدات أرقام شبه عشوائية	مولدات أرقام عشوائية حقيقية
فعالية	ممتازة	ضعيفة
حتمية	حتمية (لها سلوك متوقع)	غير حتمية (لها سلوك غير متوقع)
دورية	دورية (متكررة)	لا دورية (غير متكررة)

هذه الخصائص تجعل TRNGs مناسبة لمجموعة من التطبيقات غير المناسبة لـ PRNGs، مثل تشفير البيانات والألعاب والقمار، وعلى العكس، الفعالية الضعيفة والطبيعة غير الحتمية لل TRNGs تجعلها أقل ملائمة للمحاكاة ونمذجة التطبيقات والتي -غالبًا- ما تتطلب بيانات أكثر من توليد أرقام من TRNG.

يحتوي الجدول الآتي ملخص للتطبيقات التي تحقق أفضل النتائج حسب نوع مولد الأرقام المستخدم:

التطبيق	المولد المناسب
اليانصيب	TRNG
الألعاب والقمار	TRNG
أخذ عينات عشوائية (مثل: فحص المخدرات)	TRNG
المحاكاة والنمذجة	PRNG
الأمن (مثل: توليد مفاتيح تشفير البيانات)	TRNG
الفنون	يمكن استخدام الطريقتين حيث تتفاوت حسب النوع

الفصل الثالث

جدول (١) جدول توزيع ازمنة ما بين الوصول .

Service Time (minute)	Probability	Cumulative probability	Random number
١	0.٢٤	0.٢٤	0.000-0.٢٤0
٢	0.٣٩	0.٦٣	0.٢٤١-0.٦٣0
٣	0.٢٢	0.٨٥	0.٦٣١-0.٨٥0
٤	0.١٥	1.0٠	0.٨٥1-1.000

جدول (٢) جدول توزيع ازمنة الخدمة لعامل الاول .

Service Time (minute)	Probability	Cumulative probability	Random number
2	0.11	0.11	0.000-0.110
3	0.42	0.53	0.111-0.530
4	0.26	0.79	0.531-0.790
5	0.21	1.00	0.791-1.000

جدول (٣) جدول توزيع ازمنة الخدمة العامل الثاني .

Service Time (minute)	Probability	Cumulative probability	Random number
٣	0.٢٥	0.٢٥	0.000-0.٢٥0
٤	0.٣٩	0.٦٤	0.٢٥١-0.٦٤0
٥	0.١٠	0.7٤	0.٦٤١-0.7٤0
٦	0.٢٦	1.00	0.7٤1-1.000

No	Random Number for Arrivals	Time between arrivals	Clock time for arrival	Random number for services	Time service start	Service time	Time service end	Time service start	Service time	Time service end	Time in queue
1.	0.9481	4	4	0.0561	4	2	6				0
2.	0.8893	4	8	0.3147	8	3	11				0
3.	0.0854	1	9	0.9036				9	6	15	0
4.	0.7293	3	12	0.6492	12	4	16				0
5.	0.1878	1	13	0.146				13	3	16	2
6.	0.5268	2	15	0.1316	15	3	18				1
7.	0.7518	3	18	0.7318	18	4	22				0
8.	0.5203	2	20	0.5531				20	4	24	0
9.	0.0712	1	21	0.5919	21	4	25				1
10.	0.5069	2	23	0.0345				23	3	26	1
11.	0.6947	3	26	0.119	26	3	29				0
12.	0.2608	2	28	0.4161				28	4	32	0
13.	0.8016	3	31	0.3139	31	3	34				0
14.	0.2562	2	33	0.8533				33	6	39	0
15.	0.5638	2	35	0.812	35	5	40				0
16.	0.787	3	38	0.9344				38	6	44	1
17.	0.9369	4	42	0.3103	42	3	45				0
18.	0.7781	3	45	0.6286	45	4	49				0
19.	0.5439	2	47	0.5137				47	4	51	0
20.	0.5827	2	49	0.3887	49	3	52				0
21.	0.9539	4	53	0.1087	53	2	55				0
22.	0.9925	4	57	0.1815	57	3	60				0
23.	0.5056	2	59	0.2942				59	4	63	0
24.	0.5631	2	61	0.6553	61	4	65				0
25.	0.7081	3	64	0.9418				64	6	70	0
26.	0.1405	1	65	0.6987	65	4	69				0
27.	0.974	4	69	0.8181	69	5	74				0
28.	0.8676	4	73	0.9287				73	6	79	0
29.	0.2729	2	75	0.2483	75	3	78				0
30.	0.4474	2	77	0.1652	77	3	80				1
31.	0.0166	1	78	0.7291				78	5	83	1
32.	0.0275	1	79	0.1586	79	3	82				1
33.	0.0756	1	80	0.5153	80	3	83				2
34.	0.5715	2	82	0.5534	82	4	86				1
35.	0.6612	3	85	0.6251				85	4	89	0
36.	0.7185	3	88	0.075	88	2	90				0

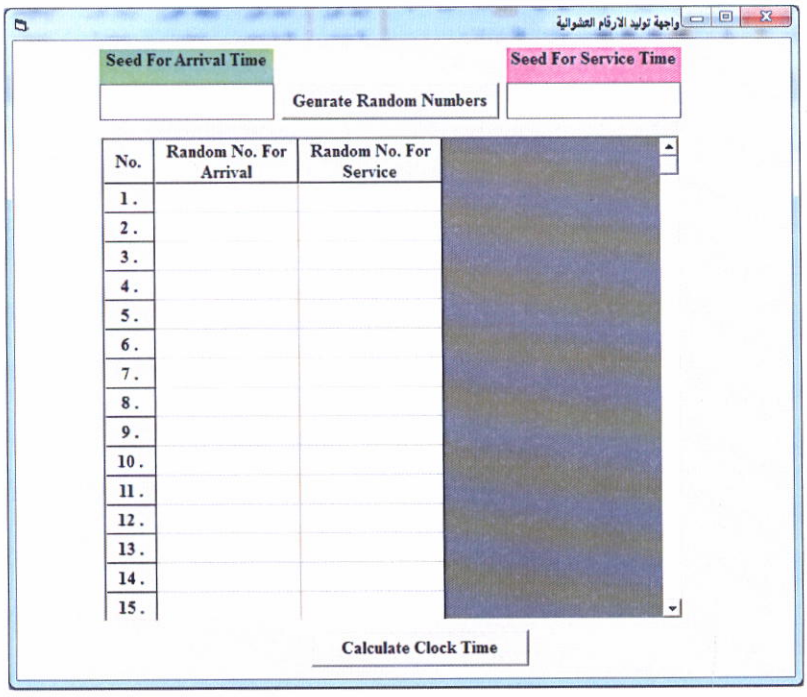
37.	0.6242	2	90	0.5625	90	4	94				0
38.	0.9625	4	94	0.6406	94	4	98				0
39.	0.6406	3	97	0.0368				97	3	100	0
40.	0.0368	1	98	0.1354	98	3	101				0

الفصل الرابع

الواجهات (الجانب العملي)

أولاً: واجهة الدخول :

وتحتوي على عنوان المشروع ، إضافة إلى اسم الطالب المنفذ للمشروع وأسم المشرف الذي قام بالإشراف عليه، وتتضمن هذه الواجهة زر أمر (دخول) وظيفتهُ الإنتقال إلى الواجهة التالية عند الضغط عليه.



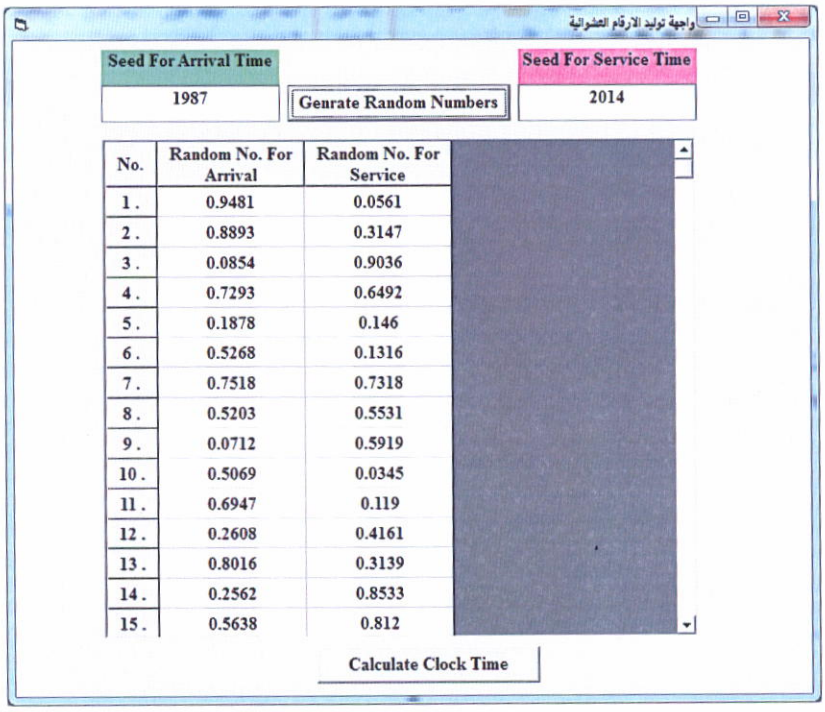
ثانياً : واجهة توليد الأرقام العشوائية:

وتحتوي على حقلين :

أ. Seed For Arrival Time : وفيه يتم إدخال البذرة لتوليد الأرقام العشوائية لأزمنة الوصول.

ب. Seed For Service Time: وفيه يتم إدخال البذرة لتوليد الأرام العشوائية لأزمنة الخدمة.

بعد إدخال الرقمين المكونين من أربع مراتب لكل رقم يتم الضغط على زر (Generate Random Numbers) لتوليد الارقام في الجدول الموضح.

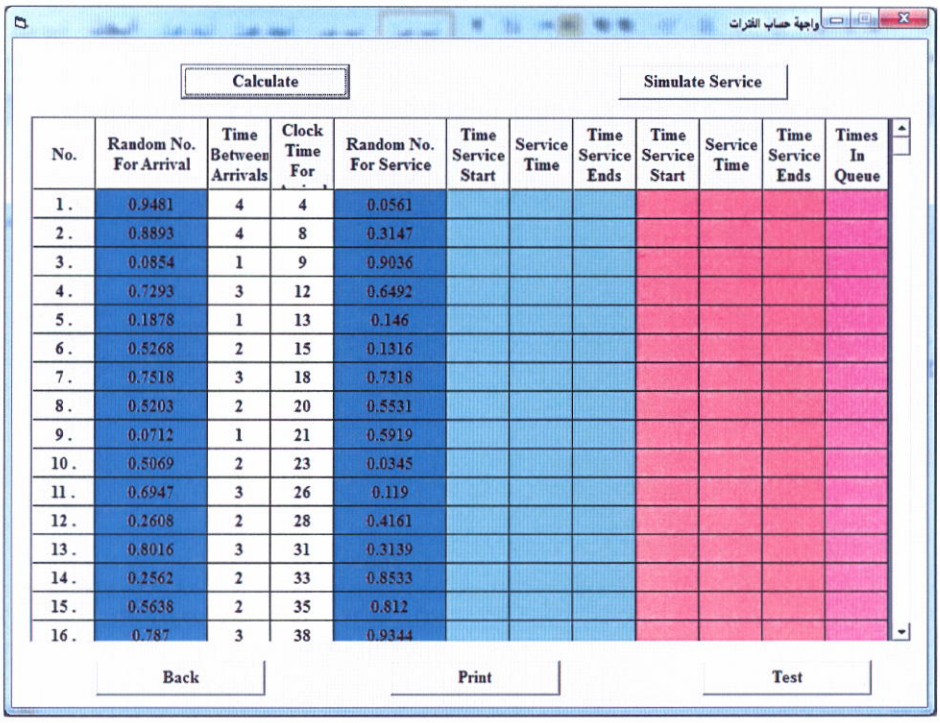


No.	Random No. For Arrival	Random No. For Service
1.	0.9481	0.0561
2.	0.8893	0.3147
3.	0.0854	0.9036
4.	0.7293	0.6492
5.	0.1878	0.146
6.	0.5268	0.1316
7.	0.7518	0.7318
8.	0.5203	0.5531
9.	0.0712	0.5919
10.	0.5069	0.0345
11.	0.6947	0.119
12.	0.2608	0.4161
13.	0.8016	0.3139
14.	0.2562	0.8533
15.	0.5638	0.812

ثالثاً: واجهة حساب الفترات:

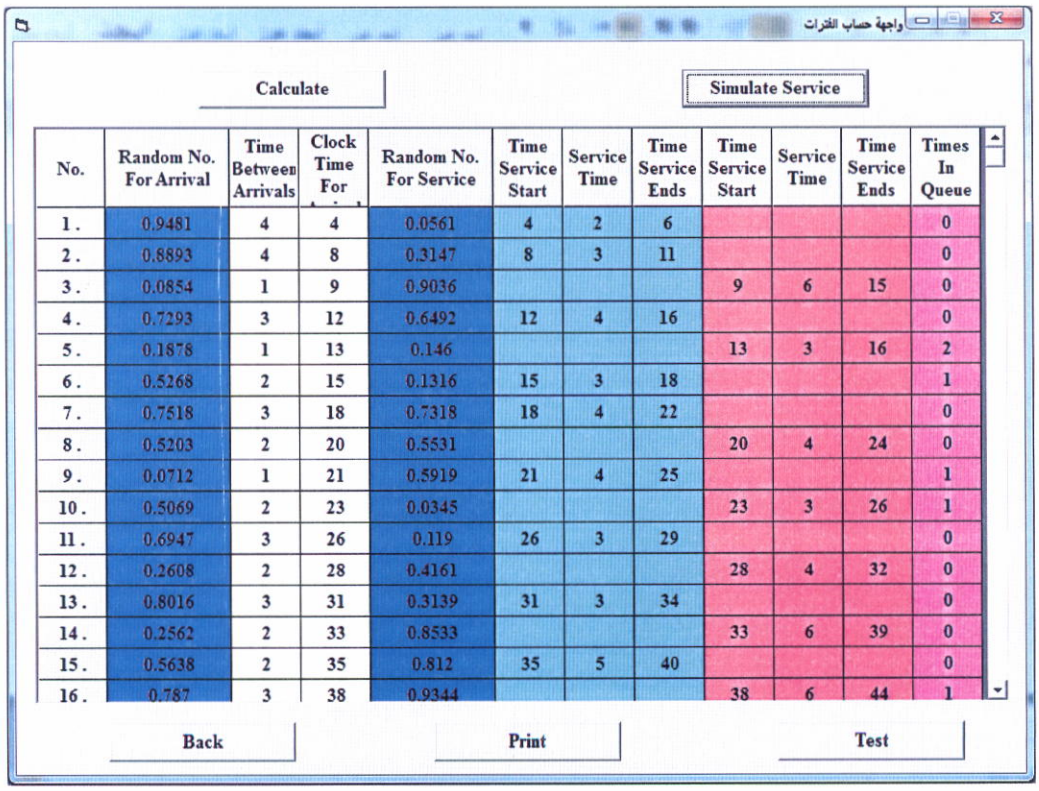
وتحتوي على أزرار الأوامر التالية:

١. (Calculate) يتم من خلاله حساب الفترات استنادا إلى الارقام العشوائية المتولدة في الواجهة الثانية :



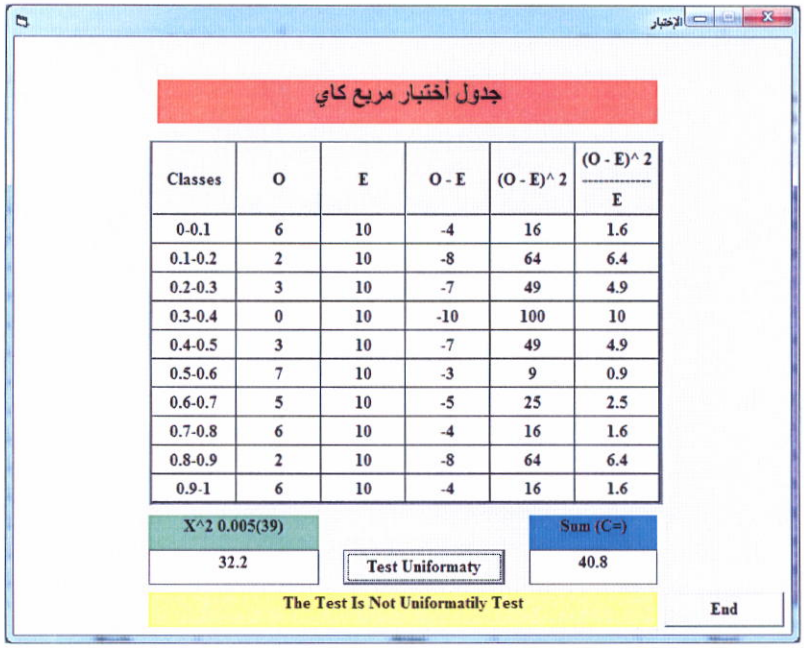
No.	Random No. For Arrival	Time Between Arrivals	Clock Time For	Random No. For Service	Time Service Start	Service Time	Time Service Ends	Time Service Start	Service Time	Time Service Ends	Times In Queue
1.	0.9481	4	4	0.0561							
2.	0.8893	4	8	0.3147							
3.	0.0854	1	9	0.9036							
4.	0.7293	3	12	0.6492							
5.	0.1878	1	13	0.146							
6.	0.5268	2	15	0.1316							
7.	0.7518	3	18	0.7318							
8.	0.5203	2	20	0.5531							
9.	0.0712	1	21	0.5919							
10.	0.5069	2	23	0.0345							
11.	0.6947	3	26	0.119							
12.	0.2608	2	28	0.4161							
13.	0.8016	3	31	0.3139							
14.	0.2562	2	33	0.8533							
15.	0.5638	2	35	0.812							
16.	0.787	3	38	0.9344							

٢. (Simulate Service) : ومن خلالها يتم احتساب أزمنة البداية والانتهاء وأزمنة الخدمة استناداً إلى الارقام المتولدة العشوائية للخدمة



No.	Random No. For Arrival	Time Between Arrivals	Clock Time For	Random No. For Service	Time Service Start	Service Time	Time Service Ends	Time Service Start	Service Time	Time Service Ends	Times In Queue
1.	0.9481	4	4	0.0561	4	2	6				0
2.	0.8893	4	8	0.3147	8	3	11				0
3.	0.0854	1	9	0.9036				9	6	15	0
4.	0.7293	3	12	0.6492	12	4	16				0
5.	0.1878	1	13	0.146				13	3	16	2
6.	0.5268	2	15	0.1316	15	3	18				1
7.	0.7518	3	18	0.7318	18	4	22				0
8.	0.5203	2	20	0.5531				20	4	24	0
9.	0.0712	1	21	0.5919	21	4	25				1
10.	0.5069	2	23	0.0345				23	3	26	1
11.	0.6947	3	26	0.119	26	3	29				0
12.	0.2608	2	28	0.4161				28	4	32	0
13.	0.8016	3	31	0.3139	31	3	34				0
14.	0.2562	2	33	0.8533				33	6	39	0
15.	0.5638	2	35	0.812	35	5	40				0
16.	0.787	3	38	0.9344				38	6	44	1

٣. (Test) : بعد أن تم توليد الارقام العشوائية يتم من خلال هذا الأمر الانتقال إلى واجهة الأختبار ليتم بطريقة (مربع – كاي) بعد إدخال قيمة الجدولية



٤. (Print) ومن خلال هذا الامر يتم تصدير النتائج إلى ملف أكسل لتوفير إمكانية الطباعة للنتائج التي أستحصلنا عليها.