Intelligent Tutor Using IBM Chatbot

Shevar Walters (1507136)

BSc in Computing, Major in Computer Science

Dejah Taylor (1507122)

BSc in Computing, Major in Computer Science

Phillip Cole (1507146)

BSc in Computing, Major in Computer Science

Kristopher Campbell (1507121)

BSc in Computing, Major in Information Technology, Minor in Networking

Project Supervisor: Sophia McNamarah

Faculty of Engineering and Computing, University of Technology, Jamaica

|  |  |
| --- | --- |
| Project Supervisor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Project Coordinator: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Table of Contents

[Abstract 5](#_Toc17792667)

[Intelligent Tutor Using IBM Chatbot 6](#_Toc17792668)

[Introduction 6](#_Toc17792669)

[Background 7](#_Toc17792670)

[Statement of the problem 9](#_Toc17792671)

[Significance of the study 9](#_Toc17792672)

[Purpose of the Study 10](#_Toc17792673)

[Delimitations 10](#_Toc17792674)

[Limitations 10](#_Toc17792675)

[Definition of Terms 10](#_Toc17792676)

[Literature Review 11](#_Toc17792677)

[Introduction 11](#_Toc17792678)

[Barriers to learning programming 12](#_Toc17792679)

[Teaching tools and learning methodologies 14](#_Toc17792680)

[Cognition and Learning Interventions 18](#_Toc17792681)

[Applications of IBM Watson 20](#_Toc17792682)

[Level 1 - Reaction 23](#_Toc17792683)

[Level 2 - Learning 24](#_Toc17792684)

[Level 3 - Behaviour 24](#_Toc17792685)

[Level 4 - Results 24](#_Toc17792686)

[Literature Review Conclusion 24](#_Toc17792687)

[Methodology 25](#_Toc17792688)

[Research Design 25](#_Toc17792689)

[Population 25](#_Toc17792690)

[Sample 26](#_Toc17792691)

[Instrument Design 26](#_Toc17792692)

[Procedure 26](#_Toc17792693)

[Data Analysis 27](#_Toc17792694)

[Project Design 27](#_Toc17792695)

[Description of Target Population 29](#_Toc17792696)

[System Evaluation 30](#_Toc17792697)

[Level 1 - Reaction 30](#_Toc17792698)

[Level 2 - Learning 30](#_Toc17792699)

[Level 3 – Behavior 31](#_Toc17792700)

[Level 4 – Results 31](#_Toc17792701)

[Results 32](#_Toc17792702)

[Introduction 32](#_Toc17792703)

[Discussion 39](#_Toc17792704)

[Conclusion and Recommendation 39](#_Toc17792705)

[Summary 39](#_Toc17792706)

[Conclusions 40](#_Toc17792707)

[Recommendations 41](#_Toc17792708)

[References 43](#_Toc17792709)

[Appendix 46](#_Toc17792710)

[SAMPLE OF EXAMINATION INSTRUMENT 46](#_Toc17792711)

# Abstract

Technology would fall within the category of a science, technology, engineering and mathematics (STEM) subject, subjects which help to improve society. Students face problems learning programming worldwide with as high as a 33% fail rate (Hoskey and Maurino, 2011). To help improve pass rate of not only Internet Technology (IT) subjects but possible other STEM subjects in Jamaica by developing an application to assist subjects in the learning of these subjects. Upon revision of the literature, coupled with the interviews conducted, the idea of using IBM Watson for the creation of an Artificial Intelligent (AI) chatbot that can handle natural language processing was decided on as the main component as well as it being equipped with interactive programming exercises. A class at the University of Technology studying Programming II was then given an exercise to test their aptitude in the subject, they then were given access to the fully developed application for 2 weeks. After this 2-week period they were given the same test once again, comparing both test results showing major improvement in pass rate compared to the first test given. At the end of using the application the students were also given a survey to evaluate the application, using these results recommendations were formulated towards an upgrade of the application. Using an application that can process natural language from student’s queries and providing an interactive learning environment, it is possible to bring about a change as to how this web-system can help with learning STEM subjects in Jamaica.

*Keywords:* Chatbot, Natural Language Processing (NLP), IBM Watson, Programming

# Intelligent Tutor Using IBM Chatbot

# Introduction

For years, educational institutions and students alike have bemoaned the difficulty in learning how to learn programming. The failure rate of students learning programming, using fundamental languages like C are staggering as mentioned by Hoskey and Maurino (2011) and stated that 33% of students fail Computer Science 1 worldwide, in addition, some students pass, but do not learn how to program. The University of Technology of Jamaica has in place a Bachelor of Science Degree in Computer programming and Information Systems for the past decades. In computing, all students are required to take two semesters of programming at an introductory level (Programming I & Programming II). Programming languages that are being taught for both courses are C and even QBasic. In addition, all students must achieve a “C-” or better in both introductory programming classes to advance into Object Oriented Programming (OOP).

According to Hoskey and Maurino (2011) it was stated that the lecturers from Farmingdale State College who are teaching these courses that students have over many years traditionally struggled with one or more of the introductory programming concepts required for success in the course. In addition, Hoskey and Maurino (2011) also stated that the biggest problem found in novice programmers is not the understanding of introductory concepts, but rather having the ability learn how to apply them.

Judging from what has been found, one can argue that the failure rate is a great deal when it comes down to programming courses which shows that students are having a difficulty in computer programming.

Despite many years of research, there is still a global problem within the educational environment relating to computer programming (Mead et al., 2006). According to Mead et al., (2006) computer science educators do not have a formal structure for defining what level of programming mastery they should expect of their students, after their first year or when they graduate. This research can help to provide a solution to this issue by improving students programming knowledge and skills using an intelligent programming tutor system to become better programmers.

## Background

An AI-based tutoring system that can process natural language from queries submitted from programming students can help improve students programming skills. IBM Watson, a cognitive technology of Artificial Intelligence in natural language processing is a step towards developing intelligent chatbot systems. The IBM Watson system can quickly use natural language processing capabilities based on what is fed into the system that can instantly retrieve relevant content in both structured and unstructured data; it tracks students’ progress, provide insights to instructors, and adapts the conversation. This would engage organizations to exploit such a system to transform industries, helping professionals do their job better, and solve important challenges out there. Watson is expected to play an important role in AI natural language processing and is the first open cognitive computing technology platform available today that entirely portrays the world in the way that humans do: through senses, learning, and experience as it continuously learns, gains in value and knowledge over time (Bloomfire, 2015).

IBM Watson, named after IBM founder Thomas J. Watson, the system was built to create a standalone platform that could rival a human’s ability to answer questions posed in natural language (Kroeker, k. L, 2011). Powered by the latest innovations in machine learning, likely application areas for Watson’s technology would be in medicine, law, education, or the financial industry.

In 2007, IBM Research took on the grand challenge of developing a computer system that could compete with two former grand champions at the game of Jeopardy (Ferrucci, D. A. 2012). On February 14-16, 2011, the IBM Watson question-answering (QA) system won against the two highest ranked players. For Watson to compete successfully against some of the game’s best former players, Watson had to answer several complex natural language questions over a broad domain of knowledge and compute an accurate confidence in its answer (which are produced from hundreds of parallel hypotheses collected and scored from evidence) as well as completing its processing in a very short amount of time. In addition, the Question-Answering (QA) problem requires the machine to go beyond just tagging and matching keywords in documents and correctly interpret the question to figure out what is being asked, it would need to find the precise answer without the help of a human to read through the returned or tagged documents (Lally & Fodor, 2011).

To solve this problem, IBM Watson research team developed a software architecture called DeepQA. The research team’s idea was that no single algorithm would accurately understand or answer all questions, therefore Watson’s intelligence was built from a wide collection of algorithms that would probabilistically and imperfectly interpret language and score evidence from different perspectives (Kroeker, 2011). It assumes and create multiple interpretations of the question, it would then create many candidate answers or hypothesis, collects contextual evidence for these hypotheses, and analyze the evidence to determine if it agree or refute those hypotheses. Watson utilizes Natural Language Processing (NLP) technology to interpret the question and analyze the vast amounts of unstructured text (encyclopedias, dictionaries, news articles, etc.) that may provide evidence in support of the answers to the questions. Some of Watson’s algorithms would then evaluate whether the relationship between entities in the question match those in the evidence (Lally & Fodor, 2011).

Furthermore, the IBM Watson cognitive system along with many other Artificial Intelligence applications such as: Nuance Communications Inc., Artificial Solutions, eGain Corporation, Creative Virtual Pvt, among others aid in the development of chatbots for natural language processing for online based services. Chatbots are predominantly classified into two categories: Scripted or Rule-based Chatbots and Artificial Intelligence-based Chatbots. AI bots are built on Machine Learning (ML) and Natural Language Processing (NLP) capabilities using Artificial Intelligence (Research and Markets, 2018).

## Statement of the problem

This research will address the problem of low pass rates in introductory programming courses due to a lack of one on one tutoring and the difficulty of grasping core programming concepts.

## Significance of the study

The research can be important in offering insights and heralding potential positive changes in the way STEM courses like computer science, specifically programming could be taught, which are of high importance in contemporary society. It can also contribute to improving the pedagogical practices of SCIT classes should the results turn out to have a resounding positive effect on the students. Students themselves may be able to benefit from the findings by having a tool that can perform automated distribution and marking of small modular programming exercises and provide access to a compilation of programming material in an AI that could help with referencing programming knowledge via natural language processing. These functionalities are intended to give them a practical refresher and tutorial on basic programming concepts which are the fundamental building blocks for being good programmers and getting better grades in the course. Since the prototype can mark the student’s assignments, it does not make the workload any heavier on lecturers and even has the potential to lighten their load. There is also a lack of any kind of automated teaching research in that is specific to Jamaica, for which our research and project could pioneer in the country and have a positive impact on the development of programmers within the university.

## Purpose of the Study

The purpose of this study is to develop an intelligent tutor system for programming. It is expected that the system will improve students’ learning as well as lessen the teacher’s workload.

## Delimitations

Our research’s focus is on the facilitation of learning using an interactive learning site. This will be accomplished using a web application as a vehicle and be targeted towards individuals learning Programming II at the University of Technology Jamaica.

## Limitations

* Because each step in a lesson is so specific, the standards for completing the step correctly are also very narrow.
* If the lesson content isn’t written carefully, students can have a hard time figuring out exactly what the lesson is asking for.
* A more serious problem with the interactive approach is that students can become accustomed to being told exactly what to do next.

## Definition of Terms

Chatbot: A computer program that accepts input from users via text or auditory channels utilizing natural language processing.

Natural Language Processing: A branch in artificial intelligence and is used to aid computers to understand the human’s natural language.

CS1: Designate the first course in the introductory sequence of a computer science major.

# Literature Review

## Introduction

Based on a study by Sánchez-Díaz, et al (2018) chatbots are computer programs designed to hold conversation with users using natural language. Some of them have human identities and personalities to make the conversation more human like. As Sánchez-Díaz, et al (2018) stated that with changing times, chatbots have grown in popularity and spread in many different industries such as education and commerce. Chatbots in the academic courses have proven to be very useful; one such example is the development of Jill Watson. Jill Watson is an intelligent tutor developed at Georgia Tech for an artificial intelligence massively open online course. The effectiveness of Watson in massive online class showed a potential of chatbots in an era where artificial intelligence will play a major role in the education system. Having a conversational bot as an intelligent tutor has some advantages such as: being available 24/7 and giving students the freedom to learn at their own pace. Having a chatbot also lightens the work load for teachers as it eases the teaching efforts of the teacher. For the beginners (learners with little or no previous knowledge), some concepts may be difficult to understand. Instead of continually applying the learned concepts in practical exercises, many beginners only do the actual practical tasks when the first formal assessment is required. In such situation the finer details of the concepts could be forgotten, resulting in their inability to successfully complete the assignment. Due to day by day increasing number of students in the class, the assessment of programming exercises leads to extensive workload for teacher/instructor, particularly if it must be carried out manually.

According to Loksa et al., (2016), “Students continue to drop out of introductory programming courses at rates of 30-50%, often because they find the material too difficult.” In this situation it is where the development of an intelligent tutor comes into place. With the implementation of an intelligent tutor, the marking of assignments can be done automatically as well as assigning with the teaching of programming. Using an intelligent tutor designed through Watson; the hope is to increase the pass rate of students in introductory programming courses as well as increase in the efficiency of assessing students’ assignments.

The following review of literature confirms the present problems of learning and assessing programming assignments, discusses specific and general solutions, and concludes that the development of an intelligent tutor system for programming is expected to improve students’ learning.

## Barriers to learning programming

Rogerson and Scott (2010) identified some of the difficulties experienced by students learning to program. The barriers to learning programming discussed provides the different teaching tools or learning methodologies which could assist or address each barrier.

Rogerson and Scott (2010) mentioned several barriers to learning programming, which include the difficulty of programming itself; learning to program often requires the student to learn new ways to think and study, the difficulty of becoming cogitative, the change and challenges of the teaching environment, and the stigma and reputation of programming. Self-awareness is an important part of learning programming. However, students may lack self-awareness because they fail to analyze the problem given and unable to solve the problems they face in a more systematic approach.

In addition, the first barrier needed to overcome for a student during their course of study is the ability to master the new way of thinking according to Rogerson & Scott (2010) in other words a student needs to able to think like a programmer. In addition to thinking like a programmer, Rogerson and Scott (2010) concludes that reflective practices increase student’s ability to understand and evaluate the programming concepts that are causing their problems and help students to become more responsible for their own learning.

In addition, the educational environment can add to the negative outcome of students not grasping the introductory programming concepts. According to Rogerson and Scott (2010) that other factors such as time constraints and large classes which can result to students not receiving immediate feedback or individual help. This can become an issue when the lack of unshared attention is reinforced with the failure to address the different learning styles of students.

Moreover, the inability to trace code is one important barrier students face when programming. As mentioned by Rogerson and Scott (2010) analyzing and having the ability to understand code, not just syntax (form) but semantics (meaning) as well requires students to be more reflective when solving any problem they are faced with.

Rogerson and Scott (2010) mentioned the stigma and reputation of being a programmer are commonly associated with being a nerd or geek creating a barrier that may prevent student programmers from fully engaging with discipline. This encourages students to not fully commit to programming and not become motivated to do more. As mentioned by Rogerson and Scott (2010) self-confidence, courage, self-esteem, persistence and self-questioning and evaluation are all important traits a student must have that will push them to increase productivity.

## Teaching tools and learning methodologies

Rogerson and Scott (2010) propose an approach to overcome these barriers to learning the aspects of programming, educators need to resolve the issue from different angles and provide ways of better teaching such as changing the course design, teaching methods and teaching tools.

According to Stanojevic and Randelovic (2017) study, it was mentioned that the web-based classroom teaching method approached used improved programming efficacy and allows students to become more attractive to the programming course. As mentioned by Regerson and Scott (2010), Educators should address the individual students and become familiar of both their knowledge level and favored learning style with programming patterns to give examples of good practice, and include games in the course, with the focus of improving problem solving abilities.

An approach that has been mentioned in many literatures that can be used by students for developing their own programs and provide an expert solution to recurring design or programming problem is with a pattern-based approach to programming instructions (Rogerson, & Scott 2010). According to Caspersen and Bennedsen (2007), students in instructional situations have shown that they choose to learn from examples rather than to learn from other forms of instructions and learn more from studying examples than from solving the same programming problems themselves. In addition, it was mentioned by Casperson and Bennedsen (2007) that beginners to programming classify problems based on the surface features of the problem statement itself, whereas professional programmers categorize problems according to features and structural similarities of their solution. Active learning techniques have been proven to have the ability to increase learning outcomes and students` attitudes towards active engagement during programming lectures mentioned by Stanojevic and Randelovic (2017).

By utilizing these necessary teaching methods and solutions to student’s programming barriers described can contribute to the improvement of student’s grades implemented through an intelligent tutoring system. As mentioned before, Casperson and Bennedsen (2007) stated that by providing example problems for students to allow them to identify the pattern to solving such problem by comparing the similarity with other problems experienced based on the problem’s features itself that can give students leverage to come up with a solution. With this pattern-based approach being implemented in the intelligent tutoring system it can help students by giving them those problem examples and along with solutions to show how to tackle such problems.

In addition, Renkl (2013) presents a theory of example-based learning that looks at three research areas, learning from worked examples, observational learning and analogical reasoning. The theory has two parts, the descriptive section, where they describe the characteristics of the three perspectives of learning such as what initial skill acquisition looks like bringing up issues such as the lack of initial understanding of a topic and the strategies a student may use to overcome the obstacle such as the use of shallow strategies like the copy and adapt strategy, where students copy a known solution and try to adapt it to another problem by only changing the variables without understanding the underlying principles required to sculpt a solution on their own.

In terms of worked examples, Renkl (2013) states in his learning approach, applying multiple worked examples alongside problems reduced the time required to learn and the number of errors made in comparison to just one example and several problems. The tutor will have no less than 2 worked examples per topic be used as a reference for the student in order to replicate these findings.

Having prompts or reminders to think about why a solution looks or works the way it does may be a key part of tutoring functionality in order to improve learning, retention and understanding of certain concepts. According to Renkl (2013), spontaneous and prompted self-explanations have been shown to lead to higher rates of knowledge transfer and application to new problems.

Example based learning is extensively studied in the field and the number of research papers supporting and augmenting this idea is numerous. Another study being done that supports this concept is the paper by Najar, Mitrovic, & McLaren (2016) which speaks to the same concept of example-based learning in an adaptive context within an intelligent tutor. The paper by Najar, Mitrovic, & McLaren (2016) showcases the practicality of the teaching method of mixing worked examples, faded examples (worked problems with some steps not visible to be filled out) and unsolved problems as a method to improve tutoring when using an intelligent tutoring system as opposed to using exclusively examples or exclusively using problems that need to be solved by the student. It highlights research that has shown that learning from examples is an effective learning strategy and noted that examples should have a step by step explanations to facilitate better learning. It notes that with the use of a not too complex algorithm, the problem solving and example giving approaches can be switched between and adapt to the student's efforts to solve a current problem placed before them and their performance based on how many steps they took to solve the problem (how many attempts they made to solve the problem).

Najer et al (2016) discusses the concept of cognitive efficiency, which is a metric that helps assess the progress of students using an intelligent tutor. It is calculated by: P divided by R where P is the number of steps required to solve the problem, in our case it would be the number of tries till they got it correct. R is the mental effort put forth by the student. A CE (Cognitive Efficiency) greater than or equal to 1 would be considered a score good enough for the student to try moving forward in the progression of tutoring. These values would be gathered from the student on each problem they did, and the number of steps would be gathered based on the log of how many attempts a student has made on a problem. The mental effort would be entered by the student, being given a choice between 1 and 10 to estimate the amount of mental effort they put into the problem.

The cognitive efficiency concept mentioned by Najar et al (2016) has proven to be useful as it offers a format that can be implemented into the learning content that the tutor comprises of, with a tested method within which to provide students with content that will stick and assist them in passing the course instead of just offering content linearly. Their research also has relevance to our project since it used experiments using an SQL Programming language tutor which is similar tutor to the C programming tutor we would like to build. This prevents us from taking a trial and error approach of developing our own adaptive model and gives us some insights into how to present the work.

Reiser, Anderson, & Farrel (1985) makes a point for the usefulness of tutoring in general, stating that private one on one tutoring is generally found to be the most effective form of instruction. Reiser, Anderson, & Farrel (1985) discussed the implementation of three pedagogical principles behind the implementation of an intelligent tutor system. The first pedagogical objective is that it ensures that the student solves problems in the domain. It ensures that the student does some actual work rather than just giving them material to read. It makes the learning experience more effective. Thus, one of the major objectives of using tutors is to make the problem-solving experience a more effective learning experience, while using the target skills to solve problems. The second pedagogical objective is that the student should do as much of the work as possible. Student learn more by doing rather than being told or reading. The tutor must ensure that exercises are available to do on based on the target skill and be able to aid when needed to limit floundering and therefore help students to solve the problem without helping them more than necessary. A third objective is that the tutor should provide immediate feedback. This should allow the student to focus on the task at hand and focus on the application of target skills rather than being consumed with the arduous task of tracking down bugs. It is more likely that a student will be able to correct the thinking or knowledge on a matter because of immediate feedback. This is because the knowledge or idea they had in mind when they made the error is still fresh in their mind when they made the error, therefore making it easier for them to correct their thinking on the spot.

A final constraint on the design of the tutor is that it should make it easier for the student to design solutions that fit the answer and solve the goals and sub-goals required to solve the problem. The structure of the solution should be provided so that the student is not left to conceptualize the solution with insufficient constraints. In the early stages of learning, they will fall back on the “generate and test” strategy which may be time consuming and imprecise.

## Cognition and Learning Interventions

Teaching the cognitive aspects of programming is one important approach to develop student’s metacognitive awareness when solving problems. There is evidence mentioned by Loksa et al., (2016) that the top performing computer science students are the ones who are aware of their problem-solving nature and are mostly capable of handling their cognitive resources. Providing contextual hints when students learn to program can help them to succeed more independently, however, it is not designed to promote metacognitive awareness.

Furthermore, there are two interventions that supports developing student’s metacognitive awareness that will directly improve student’s productivity and self-efficacy. These methods can help students towards programming while encouraging them to recognize, evaluate, and improve their problem-solving strategies.

The interventions mentioned by Loksa et al., (2016) include:

* Give clear instruction on the goals and activities in programming problem solving.
* Prompt students to describe their problem-solving state when they ask for help from a person, or from an intelligent tutoring system. This provides additional reflection on their problem solving.
* Provide a physical handout that includes the programming problem solving stages and motivates students to track which stage they are in and asking them to be aware of what actions might be appropriate next.
* Provide contextual hints that asks students to identify the problem-solving stage they are engaged in. This reinforce metacognitive awareness during code editing.

According to Loksa et al., (2016) there are six stages that can help CS students program successfully; this will teach them the psychology of programming:

* Iteratively analyze problem prompt. Programming instructions usually provide some description of a problem, which programmers must understand, interpret, and clarify.
* Search for related problems. Programmers draw upon problems that they have experienced. By solving problems to related past problems, students can better conceptualize a problem’s computational nuances.
* Search for solutions. Programmers find solutions that will solve the problem by modifying solutions they have used in the past or by finding solutions in textbooks, online, or from classmates or teachers.
* Evaluate a potential solution. Students after finding a solution must determine whether the solution will solve the problem; this includes actions like feasibility assessments, mental algorithm simulations.
* Implement a solution. With the best solution in mind, programmers must translate the solution into source code.
* Analyze implemented solution. After executing a solution, programmers iteratively test and modify the solution by evaluating how well their current implementation solves the problem.

From what was described in Loksa et al., (2016), programming requires far more than just syntax, semantics and the programming language and that it revolve around how programmers think and how they solve the problems given. Loksa et al., (2016) discussed that self-awareness is a very important attribute for students to become an expert at programming. Loksa et al., (2016) provided metacognitive interventions that can improve their self-awareness and be used to increase student’s problem-solving state. Apply these interventions within an intelligent tutoring system will become helpful to students using it and can aid in the increase of student’s grades at tertiary institutes.

## Applications of IBM Watson

In a related work, Chozas et al., (2017) used IBM Watson services that enables a dialog-based interaction with programmers during the development of programs that supports multi-platform shared memory multiprocessing programming. The description in their research identifies the use of cognitive computing to aid novice programmers learn parallel programming. Their approach was to help novice programmers to avoid commonly made mistake when they use OpenMP.

In addition, according to Chozas et al., (2017), the application design that was used was based on the IBM Watson Dialog service to provide communication interaction between the user and the system through natural language processing with the aim to avoid common OpenMP mistakes. The main purpose of the IBM Watson application is to allow the user to interact with the application using an interface, that allows the users to write questions in text form.

The functionality of the application used and developed by Chozas et al., (2017) in their work includes having an interface that sends the user input to their application which is connected to the Watson Dialog Services. The IBM Watson dialog service analyzes the questions and gathers information that is needs to provide the most suitable answer for the user. In order for Chozas et al., (2017) to use this service, the team had to first train the system by feeding IBM Watson with information that includes a list of all 32 or more common OpenMP mistakes which includes all the errors and the reasoning behind them to give the user the answer they need. The implementation of Chozas et al., (2017) proposed system was built with using xml which specifies the mandatory settings for the service which includes creating dialog tags as the main characteristic in the Dialog service and the types of folders (Main, Library, Global, and Concepts) that will be used to allocate, organize and maintain the information in the dialog.

This service, however, is now outdated as Chozas et al., (2017) mentioned that a similar service called Conversation service have been introduced, and developers are encouraged to migrate their Dialog service-based applications to the Conversation service. The retirement of this system now exists a new version of IBM Watson assistant which includes a rich set of tooling that allows users to easily create an intent classifier, extract entities, and design dialog flows with having knowledge of xml or coding.

**Automated Assessment of Student Work**

A key function of the intelligent tutor system will be the ability to automatically assess students work. To proceed in making an automatic assessment the system would need to verify how precise a student’s submitted code is compared to the given answer. Gupta, S., & Dubey, S. K. (2012) suggested an approach that used a verification program technique for the automatic assessment of programming assignments, without worry about the execution of program, input and getting output. There are three actors working in this system; teacher, student and system.

Firstly, the teacher would provide programming problems (which are presented in a descriptive format to the student), and verification program. The programming problem would be shown to the student however, verification program is hidden. Later when student visits the system, he can try to solve these problems. The works submitted by student is then assessed by the system. The stochastic information of system, such as common errors or error of program, number of inputs, number of time run would be stored in system’s database, which can help the teacher to evaluate the performance of the students and overall performance on the whole course. This automatic assessment system automatically compiles and runs the student's program and evaluates it based on the verification program. This system gives the result of assessment in less effort and provides all information regarding programming assessment.

Rahman, et al (2016) suggests that when it comes to designing a system for automated assignment two approaches are possible (1) A dynamic approach and (2) A static approach. A dynamic approach completely executes the program and then checks the program’s correctness. The static approach assesses the program without it being executed. Rahman went about it in a static approach by taking both the instructor’s and student’s source code and converted it into pseudo-code for comparison, to find the similarity percentage.

As to how his system worked there were two user categories; student and instructor. The instructors would prepare all the questions and grant the students access to the system during their sessions. For each question that is prepared the instructor also prepares all possible answers. With the use of multiple answers, it gives more room for freedom in the student’s case when submitting answers so that there is better similarity percentage between the student’s answers and the schemes provided. Utilizing this basic approach for the intelligent tutor to begin the process intelligent tutor will accept the student’s source code and translate it into pseudo-code. The same goes for all the code schemes provided by the instructor. The two codes will then be compared, and the similarity percentage stored. The process will then repeat with the same student’s pseudo-code compared alongside all the instructors pseudo-code scheme. Due to the outcome percentage of each pseudo-code will differ, the highest percentage is chosen to be the mark for the student’s answer. The mark will be stored and then displayed towards the instructor.

The intelligent tutor is meant to help students acquire new programming skills, but the question is how to measure the effectiveness of our application for students. Donald Kirkpatrick (2007) suggested a method to help training developers and HR specialists measure the effectiveness of their training initiatives. He developed four levels that are designed as a sequence of ways to evaluate training programs. The four levels of Kirkpatrick's evaluation model are as follows:

### Level 1 - Reaction

This is what participants thought and felt about the training (satisfaction; "smile sheets")

### Level 2 - Learning

This is the resulting increase in knowledge and/or skills and change in attitudes. This evaluation occurs during the training in the form of either a knowledge demonstration or test.

### Level 3 - Behaviour

This determines the transfer of knowledge, skills, and/or attitudes from classroom to the job (change in job behaviour due to training program).

### Level 4 - Results

These are the results that occurred because of attendance and participation in a training program (can be monetary, performance-based, etc.)

## Literature Review Conclusion

The literature reviewed offers some general insights into the construction of our tutor. The important takeaways from the literature are that examples and exercises are important as it allows to shorten training time. The literature also states that the student should think through their own cognition of the problem by reflecting on why a solution makes sense to be able to form connections to the solutions of their own accord in order to improve understanding and retention. Research above also recommend providing contextual hints on the problem so that students learn to better identify the type of problem they are working on. Papers also show further evidence of tutors being used in education such as when Georgia Tech implemented an IBM Watson chatbot in their Open Online Courses.

Regarding gaps in existing knowledge, there seems to be a lack of research like this in the country or even in the region, so some of the applications from our findings may not be as applicable to Jamaica in the Caribbean region. More research and exploration of possibilities will be required to acquire a better understanding of why and how computerized tutoring can assist with learning subjects such as programming in comparison to leaving students to their own devices and to what extent does it improve or worsen their learning, particularly in Jamaica.

# Methodology

## Research Design

The research design is that of a quantitative category which will ensure that the problem is effectively addressed and analyzed as objectively as possible. According to the online business dictionary quantitative research is the use of sampling techniques whose findings may be expressed numerically and are amenable to mathematical manipulation enabling the researcher to estimate future events or quantities. This type of research is the basis for the collection, measurement and analysis of data. The research is centered around the improvement of programming efficiency in students. The quantitative approach will allow for the collection of statistical and numerical data to explain, describe, predict or control the phenomena being studied.

Experimental Design is the research design our group will be undertaking. This type of research is chosen because it allows for the researcher to measure the effects of the independent variables (Our AI Tutor) on dependent variables, which is effectiveness of learning and implementing C Programming. The student’s learning and improvement will be analysed based on how well they were able to perform on a multiple-choice test in order to evaluate the dependent variable, the ease of learning on the application and the correctness of their responses.

## Population

The targeted population for this study is the students at the University of Technology, Jamaica, within the module of Programming II. The students should be within the age of 17 – 30 and the population of students currently in Programming II is approximately 30 students. Individuals that fall within these characteristics will be asked to volunteer for the experiment by asking the class at the end of a lecture or asking a lecturer to relay the request to them.

## Sample

The sample size of this research design will be 30 students from a Programming II class. This sample was chosen through the random sampling method where students that were interested volunteered themselves to use the prototype application and fill out questionnaires that tested whether they improved their knowledge in key topics such as structures and files.

## Instrument Design

The instrument we will use is a programming test, in order to measure the impact of the intelligent tutor on their studies. A multiple-choice exam has been designed with topics that the student should be learning in Programming II. The instrument will span topics such as: Functions, Sequential files, Arrays, pointers, and Structures.

There are only twelve multiple choice questions in order to maximise the convenience of the volunteers. This means the test will be marked out of twelve marks. The higher the amount a student has correct, the higher their performance.

## Procedure

Data was collected over a span of approximately three weeks. In the first week, 30 participants were given a quiz to test their programming knowledge. They were then given the web app to use for 2 hours per day for two weeks following the Programming II syllabus throughout the duration of this research with the aim to teach concepts, syntax and semantics of Programming II in C. The participants were then at the end of the week were given the quiz once again to measure the students programming ability at the University of Technology. Since this research involved recording the improvements of student’s grades, students registered for Programming II course best suited to participate in the study.

A survey was distributed to 30 participants, who consists of students currently doing Programming II as well as students who did Programming II last semester to provide feedback of the web application used for learning programming in C.

## Data Analysis

As an experimental research for Programming II classes, one change added to the group after testing the students initially; the ten 2-hour weekday sessions using the intelligent tutoring system to examine the improvements in students’ grades. In this section, we describe the changes these additions caused, beginning with a qualitative description of the students’ experiences and outcomes to give context to our results and feedback of the overall system use. We utilize majority rule to represent the quality of the data collected.

## Project Design

The project design is based on an online web-based application that utilizes IBM Watson Assistant service to provide communication means between programming students and the system through natural language processing with the aim to improve the learning aspects of programming.

Firstly, a registered account is needed on IBM Bluemix to accomplish access to the powerful IBM Watson platform. A Dialog Skill needs to be created which understands typical questions or requests from users and answers or fulfills them by following a dialog that is scripted. A dialog skill contains the training data and logic that enables an assistant to help customers. It contains the following types of artifacts: Intents, Entities and Dialog. Intents, Entities are designed based on the syllabus of Programming II and the Dialog which will give answers to the user’s questions is trained with several unstructured text & data (images, encyclopedias) and the use of a C programming textbook “C How to Program by Harvey Deitel and Paul Deitel”. The Dialog is capable of providing a response to students in both text and imagery depending on the student’s query, for example, if the student chose to know about pointers, the student may want to know how to make pointers in C, the tutor would give an example in perhaps in text format with visual aids to aid in understanding.

The goal would be to include our trained Watson Assistant Model using node.js to perform the required API calls to send messages back and forth between the application and our IBM Assistant model on IBM’s servers.

We would need to design exercises that test and trains the student within the confines of the Programming II syllabus. These exercises would consist of mostly fill in the blank type questions that focuses on the myriad of situations each topic in Programming II could be applied to.

We would also insert into exercises and examples, reminders or pop ups that encourages meta cognitive thoughts. An example of this would be, during an example of traversing through an array of arrays, a context box would pop up with tips on how to visualize what an array of arrays would look like and ask the student to think about why the indexes operate the way they do.

We would need to create a user interface that allows the student to choose whether they wish to interact with the intelligent aspect of the tutor, to interact and query concepts from the syllabus, using natural language to do the queries.

## Description of Target Population

The targeted population for this study is all the students at the university of technology within the module of Programming 2 for the semester of January 2019 to May 2019. The students would have done and passed Programming 1 which is a pre-requisite module to Programming II. Some of the specific benefits of online learning include:

1. Allows students to learn at their own pace

Students’ concentration levels vary widely, meaning some students are fast learners while others are not. These variations in speed can affect slow learners' motivation if the teacher does not take a leading role to help the student. Dealing with a slow learner calls for special efforts from an educator. Otherwise the student will fall behind in school. With a chatbot, students learn at their own speed, within the learning schedule set by their educator.

1. Convenience

One of the benefits of learning online you will hear is convenience. Rather than having to be in the same room with the teacher, learning can be done at home and if the student possesses a portable device like a tablet or a laptop, then learning can take place almost anywhere.

1. Join virtual study groups

It's possible to form study groups online using the chatbot to study remotely with others. All they need is a computer, a headset to listen and talk in, and the time to join others virtually to swap ideas about classroom assignments, get questions answered about things their stuck on. They can share screens, present documents, use virtual whiteboards to brainstorm ideas and organize their study projects.

1. Review course materials repeatedly

Unlike classroom teaching, with online learning you can access the content an unlimited number of times. Even if it's 3AM, students have access to online course materials such as videos, podcasts, written materials to reinforce course concepts and theory they might be stuck on. It's so much easier to flip through screens, use the "find" function to quickly zero in on relevant paragraphs quickly and take online notes to help them understand the class materials.

1. Assistance without your professor

It's not always convenient to see a college course instructor face to face, so the chatbot presents more opportunities to get feedback or have a great Q&A session without your college professor.

## System Evaluation

Donald Kirkpatrick (2007) suggested a framework for evaluating effectiveness of training. We will use this framework to evaluate the effectiveness of our system.

### Level 1 - Reaction

We would use feedback from every student after completing the course where they rate us on a 5-star scale. Feedback would be based on overall experience as well as individual components such as videos, exercises and support to help us get more granular data on what is working and what isn't.

### Level 2 - Learning

To measure this, we would compare their test score and sample test score that every student acquires. So, if you scored 30 in a pre-test and 70 in final test, this means training helped you bridge 40/70 i.e. ~57% of the gap.

### Level 3 – Behavior

The course already has a project that a student is expected to work on independently and submit during the course. For each project problem statement, a rubric is developed which would have various parameters with different weights against which each submission will be evaluated and given a final score. By comparing the different weightings from the rubric for each student and their final grade we can assess how effectively students have applied the concepts taught.

### Level 4 – Results

To measure results a pre and post survey of expectations was developed. In pre-survey, we ask a student what the result she expects after completing the training (this could range from 'good marks in exam', to 'build a project for self' etc.). We reach out to same set of students after the course and ask them to what extent they were able to achieve the results they had expected and their opinions on what could be changed or implemented to improve the course.

# Results

## Introduction

The purpose of this study is to develop an intelligent tutor system for programming. It is expected that the system will improve students’ learning as well as lessen the teacher’s work load.  Using a survey distributed to 30 participants, who consist of students currently doing programming II as well as students who did programming II last semester, the compiled results are as followed.

Figure 1. An evaluation on how easy for users to navigate the web-application.

The pie chart shows the representation of what the participants thought of the website navigation. Showing only a three percent (3%) of a negative review towards the site’s navigation while the other seventy-four (74%) being positive and a percentile of 23 for neutral.

Figure 2. Programming topics that students find challenging.

Figure 2 shows the most difficult programming concepts that the participants have attempted to learn. The areas that seemed most difficult were Pointers and references with thirty four percent (34%) of respondents regarding it as difficult, with Recursion in second with twenty-three percent (23%) and a for third place with thirteen percent (13%) was Loop Structures.

Figure 3. The web-application evaluation from users visit.

Figure 3 is a representation of the data to see how well the website meet the participants standards. The results show that most respondents were satisfied with what they were presented with a total of sixty three percent (63%).

Figure 4. Students review on how well content is organized on the web-application.

Figure 4 shows participants opinion on how well the information is organized. Sixty three percent (63%) of the participants agree to the information being well organized.

Figure 5. Students satisfactory evaluation of the web-application.

Figure 5 shows how the participants rate the web site based on all aspects of it. With a total of twenty three percent (23%) of participants saying that the web app was poor, seventeen percent (17%) saying it was fair, thirty three percent (33%) rate it as good while twenty seven percent (27%) rate it as excellent.

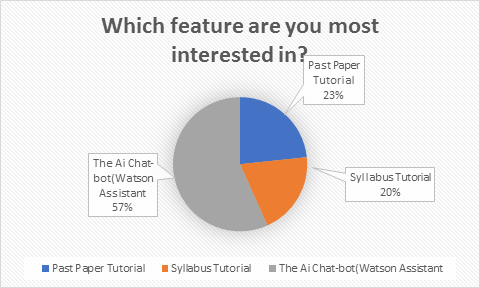


Figure 6. The feature the students are mostly interested in from the web-aplication.

Figure 6 shows the tallied most interesting feature to the participants. The most interesting feature present on the web app would be the AI Chatbot (Watson Assistant) with a total of fifty seven percent (57%) of responses and the less interesting being the Syllabus Tutorial with a twenty percent (20%) response rate. In between both of those responses at twenty three percent (23%) was the Past Paper Tutorial feature.

Figure 7. The feature students are least interested in on the web-application.

In figure 7 we collected data on which feature was used the least, with Syllabus Tutorial having the highest percentage at sixty seven percent (63%). Behind that result the AI Chatbot (Watson Assistant) has a percentage of twenty seven percent (27%) and lastly the Past Paper Tutorial has a ten percent (10%) response rate making it the most underutilized feature of the web app.

Figure 8. Students general interest of the web-application.

The above chart shows the general interest of using a web-app tutor equipped with AI to aid in the learning concepts of programming II with a majority saying yes with a sixty percent (60%) response rate.

Figure 9. The interest of continued use of the web-application from students.

Figure 9 shows the interest in the continued use of the web app. Forty two percent (42%) of participants say very likely with thirty seven percent (37%) saying somewhat likely and a response rate of twenty percent (20%) saying that it would be Not Likely.

Figure 10. Before and after statistics of students improvement from using the web-application.

Figure 10 depicts the results of people taking the same quiz. Before and after using the web app. It shows a total number of nine (9) people passing the quiz before the use of the web app. The thirty (30) participants were given a quiz to test their programming knowledge, which only nine (9) participants passed. They were then given the web app to use for 2 hours and were administered the quiz once again, this time however there was a hundred percent (100%) pass rate on the quiz, meaning all thirty (30) participants passed.

# Discussion

The majority of participants welcomed the use of a web app indicated by respondents as shown in figures 8 and 9. Also based on the feedback from respondents the overall website design and premise was to acceptable standard as the results show in figures 3,4,5 and 10. The most difficult topics include pointers, references and recursion. The most useful feature in the web app was the AI Chatbot (Watson Assistant). Knowing the most difficult topics paired with the most useful feature, more help can be imparted to focus on those topics within the AI Chatbot (Watson Assistant). The least used feature available on the web app would be the syllabus tutorial, which includes the syllabus, lectures and example tutorials, this will be taken into future consideration for adjustments to provide more use for it.

# Conclusion and Recommendation

## Summary

Throughout this study, the problem was the low pass rates in introductory programming courses due to the difficulty of grasping core programming concepts. To solve this problem, an intelligent tutor system for programming was developed. Based on survey findings and test results the system seem to improve students’ learning as well as lessen the teacher’s workload.

The intelligent tutor web app that was created was written using PHP and JavaScript and using the Watson Assistant as its core function. Watson Assistant which serves as a chatbot was trained with several unstructured text & data (images, encyclopedias) and textbooks to provide evidence in support of the answers to the questions that students may ask relating to Programming II. The web app includes an online tutoring feature that students can write code to particular exam programming questions in C. This feature essentially acts as a compiler, examining the correctness of the C code produced by the student to a given programming problem.

The final feature in this system is the ability for lecturers to be able to update or insert a new batch of programming II past papers for students to work on, exploiting this feature further on may allow this system to become scalable to other subjects. A quiz was administered to 30 students who were currently doing Programming II or did it the previous semester. The results were collected, and the participants were unaware of the grade they received. Participants were then allowed to use the web app for up to 2 hours after which they were given the same test again. The pass rate increased from the first test to having only a minimum number of passes to a 100% pass rate from the participants.

## Conclusions

In this research paper we have investigated the use of a C programming focused web application equipped with a cognitive computing service to help programmers learn and become better at coding in C. We have used the IBM Watson Assistant service to establish the interaction between the user and the system in a chatbot fashion using natural language.

The Chatbot application is trained to respond to questions related to Programming II concepts using the University of Technology Jamaica Programming II syllabus as a guide for those answers along with a C programming book called “C How to Program by Harvey Deitel and Paul Deitel”. The chatbot help with concepts understanding with visual illustrations such as images and moving images to increase students’ metacognitive abilities, and present code that produces the correct result.

In addition, a system like this may reduce the time investment required to learn C programming. The proposed tool can answer questions that a user may ask, it can’t however generate code, compile code nor find code errors. To evaluate the usefulness of this application, we conducted a survey with several both novice and experienced C programmers. The results of the survey showed that novice C programmers are willing to use such interactive application while writing their C programs and in addition, would find it useful to improve their knowledge of the subject and programming concepts.

The results show that users are more than likely to revisit the web-application. Of a set of 30 students only 9 of them passed the test with a score of over 50%, after using the application for a testing period, all 30 students were able to get a passing score on the test. The application can accomplish the goal we set out for it to do, but it shows that this method of learning has potential and is capable of being built upon to get something really useful, but more rigid tests and further development and polish of the application is recommended.

Furthermore, we believe that our tool can be used as an educational resource in a beginner’s course in C programming for Programming II and hopefully become scalable enough to be used for other programming modules that Utech offers. Future research will address the extension of our solution with additional IBM Watson services that enable automatic input of data from existing sources of C programming knowledge with possibly new features added to the main system.

## Recommendations

The Watson assistant, when offering information regarding the topic requested, could also simultaneously offer the tutorial questions for them to answer at the same time. This could speed up the learning process and allow them to apply the information faster.

Hints and tips could pop up during the use of the Watson assistant and the tutorial answering aspects of the application. In our literature review, we learned about the possibility of metacognition, the concept of allowing the user/ student to think about the information that they are learning and applying in the tutorial/ quiz section will allow them to absorb the information much faster.

The proposed tool can answer questions that a user may ask, however it will neither generate code nor find code errors, therefore the main system currently will do code compilation and checking. In addition, code generation of simple algorithms for users with the intent to generate and do a particular task and code debugging activities are additional future interest for the main web System.

For future ventures of this cognitive system, we hope that machine learning can play a part of Watson so that it can learn from the users collected input allowing the application to be improved as it is used possibly with the use of IBM cloud services such as Watson Studio and Machine Learning.

References

Bloomfire. (2015). Bloomfire Taps IBM Watson to Enhance Searchability. Business Wire (English). Retrieved October 21, 2016 from <http://search.ebscohost.com.ezproxy.utech.edu.jm/login.aspx?direct=true&db=bwh&AN=bizwire.c64284109&site=ehost-live>

Caspersen, M. E., & Bennedsen, J. (2007, September). Instructional design of a programming course: a learning theoretic approach. In Proceedings of the third international workshop on Computing education research (pp. 111-122). ACM. Retrieved January 4, 2019

Chozas, A. C., Memeti, S., & Pllana, S. (2017). Using Cognitive Computing for Learning Parallel Programming: An IBM Watson Solution. *arXiv preprint arXiv:1704.01513*. Retrieved: November 14, 2018.

Ferrucci, D. A. (2012). Introduction to “this is watson”. IBM Journal of Research and Development, 56(3.4), 1-1.

Gertner, J. (2012). Calling Dr Watson. Fast Company, (170), 124–141. Retrieved October 21, 2018 from <http://search.ebscohost.com.ezproxy.utech.edu.jm/login.aspx?direct=true&db=bth&AN=82544516&site=ehost-live>

Gupta, S., & Dubey, S. K. (2012). Automatic Assessment of Programming assignment. *Computer Science Conference Proceedings*. Retrieved November 01, 2018.

Hoskey, A., & Maurino, P. S. M. (2011). Beyond introductory programming: Success factors for advanced programming. Information Systems Education Journal, 9(5), 61.

Kroeker, K. L. (2011). Weighing Watson’s Impact. Communications of the ACM, 54(7), 13–15. <https://doi.org/10.1145/1965724.1965730>

Lally, A., & Fodor, P. (2011). Natural language processing with prolog in the ibm watson system. *The Association for Logic Programming (ALP) Newsletter*.

Loksa, D., Ko, A. J., Jernigan, W., Oleson, A., Mendez, C. J., & Burnett, M. M. (2016, May). Programming, problem solving, and self-awareness: effects of explicit guidance. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (pp. 1449-1461). ACM. Retrieved: November 1, 2018.

Mead, J., Gray, S., Hamer, J., James, R., Sorva, J., Clair, C. S., & Thomas, L. (2006). A cognitive approach to identifying measurable milestones for programming skill acquisition. ACM SIGCSE Bulletin, 38(4), 182-194.

Najar, A. S., Mitrovic, A., & McLaren, B. M. (2016). Learning with intelligent tutors and worked examples: selecting learning activities adaptively leads to better learning outcomes than a fixed curriculum. *Springer Science*. Retrieved November 11, 2018.

Rahman, Khirulnizam & Ahmad, Syarbaini & Nordin, Md Jan. (2007, January), The Design of an Automated C Programming Assessment Using Pseudo-code Comparison Technique. Retrieved: October 25, 2018, from <https://www.researchgate.net/publication/228957435_The_Design_of_an_Automated_C_Programming_Assessment_Using_Pseudo-code_Comparison_Technique>

Reiser, B. J., Anderson, J. R., & Farrel, R. G. (1985). DYNAMIC STUDENT MODELLING IN AN INTELLIGENT TUTOR FOR LISP PROGRAMMING . *IJCAI*. Retrieved November 11, 2018.

Renkl, A. (2014). Toward an Instructionally Oriented Theory of Example-Based Learning. *Cognitive Science*, 1-37. Retrieved November 11, 2018.

Research and Markets. (2018). Global Chatbots Market Outlook to 2022: Major Players are IBM Watson, Nuance Communications, Artificial Solutions, eGain & Creative Virtual - ResearchAndMarkets.com. *Business Wire (English)*. Retrieved October 23, 2018 from <http://search.ebscohost.com.ezproxy.utech.edu.jm/login.aspx?direct=true&db=bwh&AN=bizwire.c82775886&site=ehost-live>

Rogerson, C., & Scott, E. (2010). The Fear Factor: How It Affects Students Learning to Program in a Tertiary Environment. Journal of Information Technology Education, 9, 147–171.

Retrieved: November 1, 2018 from <https://doi.org/>.

Sánchez-Díaz, Xavier & Ayala-Bastidas, Gilberto & Fonseca-Ortiz, Pedro & Garrido, Leonardo. (2018,September), A Knowledge-based Methodology for Building a Conversational Chatbot as an Intelligent Tutor. (2018, September). Retrieved November 1, 2018, from https://www.researchgate.net/publication/327391803\_A\_Knowledge-based\_Methodology\_for\_Building\_a\_Conversational\_Chatbot\_as\_an\_Intelligent\_Tutor

Stanojević, L., & Ranđelović, M. (2017). THE EFFECT OF WEB-BASED CLASSROOM RESPONSE SYSTEM ON STUDENTS LEARNING OUTCOMES: RESULTS FROM PROGRAMMING COURSE. *International Conference of Information Technology and Development of Education*. Retrieved October 22, 2018.

Watson, C., & Li, F. W. (2014). Failure rates in introductory programming revisited [Abstract]. Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education - ITiCSE 14. doi:10.1145/2591708.2591749

Woszczynski, A. B., Guthrie, T. C., & Shade, S. (2005). Personality and Programming. *Journal of Information Systems Education*, *16*(3), 293–299. Retrieved October 22, 2018 from <http://search.ebscohost.com.ezproxy.utech.edu.jm/login.aspx?direct=true&db=bth&AN=18329761&site=ehost-live>

# Appendix

## SAMPLE OF EXAMINATION INSTRUMENT

Answer all the questions as quickly and as accurately as you can. Choose the correct answer from these multiple-choice questions.

1. **In C, parameters are always**

|  |  |
| --- | --- |
| A. | Passed by value |
| B. | Passed by reference |
| C. | Non-pointer variables are passed by value and pointers are passed by reference |
| D. | Passed by value result |

1. **Which of the following is true about return type of functions in C?**

|  |  |
| --- | --- |
| A. | Functions can return any type |
| B. | Functions can return any type except array and functions |
| C. | Functions can return any type except array, functions and union |
| D. | Functions can return any type except array, functions, function pointer and union |

1. **Which of the following true about FILE \*fp**

|  |  |
| --- | --- |
| A. | FILE is a keyword in C for representing files and fp is a variable of FILE type. |
| B. | FILE is a structure and fp is a pointer to the structure of FILE type |
| C. | FILE is a stream |
| D. | FILE is a buffered stream |

1. **What is the purpose of "rb" in fopen() function used below in the code?**

FILE \*fp;

fp = fopen("source.txt", "rb");

A. open "source.txt" in binary mode for reading

B. open "source.txt" in binary mode for reading and writing

C. Create a new file "source.txt" for reading and writing

D. None of above

**5. If a variable is a pointer to a structure, then which of the following operator is used to access data members of the structure through the pointer variable?**

A. .

B. &

C. \*

D. ->

**6. How will you print \n on the screen?**

A. printf("\n");

B. echo "\\n";

C. printf('\n');

D. printf("\\n");

**7. What will be the output of the program?**

#include<stdio.h>

#include<string.h>

int main()

{

char str1[20] = "Hello", str2[20] = " World";

printf("%s\n", strcpy(str2, strcat(str1, str2)));

return 0;

}

A. Hello

B. World

C. Hello World

D. WorldHello

**8. What will be the output of the program?**

#include<stdio.h>

#include<string.h>

int main()

{

printf("%d\n", strlen("123456"));

return 0;

}

A. 6

B. 12

C. 7

D. 2

**9. Point out the error in the program**

struct emp

{

int ecode;

struct emp \*e;

};

A. Error: in structure declaration

B. Linker Error

C. No Error

D. None of above

**10. A union cannot be nested in a structure**

A. True

B. False

**11. A structure can be nested inside another structure.**

A. True

B. False

**12. Point out the error in the program**

f(int a, int b)

{

int a;

a = 20;

return a;

}

A. Missing parenthesis in return statement

B. The function should be defined as int f(int a, int b)

C. Redeclaration of a

D. None of above