**Course: Robotics & Mechatronics T**

**Unit: Building & Programming Circuits**

**Unit Value: 1.0 Semester 1, 2020 Assessment Value:** XX%

**Assessment Item:** Project **Teacher**: Mr C. Johnson

**Date Notified**: XXXXXXXXXXX (Week XX)

Draft Due: XXXXXXXXXXX (Week XX)

**Date Due:** XXXXXXXXXXX (Week XX)

**TASK SUMMARY:**

It can be helpful for some people to reduce their screen time, or to know how often they are accessing their phone. In this project you will build a simple mobile phone base, which can be monitored by a BBC micro:bit, to determine whether the phone is on the base or not. You will program the micro:bit to keep a log of when the phone is present on the base or not, with dates and times, storing this data in a file on the micro:bit. You will also program the micro:bit with another extension feature of your choice, chosen in consultation with your teacher.

You will submit a written report documenting your solution, and also a short video as evidence of your working solution.

Work submitted should not be hand-written, including text, diagrams, equations, tables, graphs. “Camel case” should be used in any produced code for all variable and function names. Appropriate computer applications should be used to produce you work. As an exception, complex diagrams may be hand drawn if they are difficult to produce with a computer, but they should be scanned or photographed for insertion into your work.

**NOTE**: This assignment is to be your own individual work, and there should be no collaboration between students. You may consult with your teacher if you are uncertain about what is required.

*Drafts will not be accepted after the due date above. The draft is for general feedback only and will not be marked and graded, and marks will be based only on the final submission.*

**Submission Instructions:**

1. All work needs to be submitted electronically via Schoolbox by 1.15 pm on the day it is due, including a copy of the College Assessment Task Cover Sheet (including the Declaration of Original Work). **Problems with printing or associated computer/internet problems are not regarded as valid reasons for late submission of work.**
2. For clarification on BSSS Policies on penalties for late submission and plagiarism where work is completed out of class, students should refer to one or more of the following; Unit Outlines, the BCC Student Assessment Booklet, the BCC Year 11 and 12 Student Handbook, the BSSS publication, ‘What’s Plagiarism: How you can avoid it” and the BSSS Website <http://www.bsss.act.edu.au/The_Board/policy_and_procedures_manual>.

**TASK DETAILS:**

You are to create the following solutions:

1. Build a simple mobile phone base that can be used by a BBC micro:bit to detect whether a mobile phone is present on the base or not. It should be a simple mechanical device (ie no active components), that has a mechanism which either opens or closes a circuit when the phone is removed from the base. The base will connect by wires to the micro:bit, which will use the pin is\_touched() function to detect if the circuit is open or closed (it will be up to you whether the circuit is open when the phone is present or closed when it is present). The base need only be constructed of cheap and readily available materials, such as cardboard, rubber bands and aluminium foil. You may also use other freely available recycled waste materials. You do have to design and build something yourself, not buy something made-to-purpose. *NOTE: You will not be marked on how well it is designed, or what kinds of materials you use. It simply needs to perform one function, of either opening or closing the circuit, depending on whether the phone is present or not. So do not spend too much time making your base!*
2. Program the BBC micro:bit using Python to perform the following functions:
   1. When it is turned on, ask the user to set the current date and time
   2. Keep track of the date and time, updating it every second, and storing values in separate variables for each of the following:
      1. Date of the Month (ie 1 to 31)
      2. Month number (ie 1 to 12)
      3. 4-digit Year number (eg 2020)
      4. Hours number in 24-hour time (ie 0 to 23)
      5. Minutes number (ie 0 to 59)
      6. Seconds number (ie 0 to 59)
   3. Create a log file on the micro:bit called “log.txt” which has the following format for each line:  
        
      YYYY-MM-DD,HH:MM:SS,present/away  
        
      When the micro:bit starts, an initial entry should be recorded, as well as an entry each time the phone is removed or replaced. For example, entries may look like this:  
        
      2020-06-03,07:30:23,present  
      2020-06-03,07:58:41,away  
      2020-06-03,16:02:05,present
   4. Check in on the base once every 5 seconds to determine if the phone is present or away.
   5. When the phone is present on the base, the micro:bit LED display should usually be blank. When the phone is away, the LED display should display Image.SAD for 1 second every five seconds (ie off for four seconds).
3. In consultation with the teacher, program the micro:bit using Python with one additional feature from the list below, or another feature negotiated with the teacher (each student must choose a different feature):
   1. **Light Sensor**: use in-built sensor to determine whether it is light or dark and add either ‘light’ or ‘dark’ at the end of each log entry. Also play audible noise if the phone is removed in the dark.
   2. **Timer**: when the phone is removed and button B is pressed, set a timer for 5 minutes, 10 minutes, 15 minutes, 20 minutes or 30 minutes. Play an audible alarm after the chosen time elapses, and repeating for each interval period the lapses if the phone isn’t returned. Also add to the log entry the interval time selected if one is selected when the phone is removed.
   3. **Best and Worst**: Keep a record of longest time phone is present (“best”) and away (“worst”). If the time is exceeded, play an appropriate sound, and also display an appropriate message for two minutes. While the phone continues to be present after a new “best” time, re-play the message every 30 minutes. Every 15 minutes that the phone continues to be away after a new “worst” time, re-sound alarm and scroll a message.
   4. **Score**: Keep a track of total times present and away since starting, as a number of days, including decimal places for part of a day. If button B is pressed, display a percentage representing the proportion of whole time that the phone was present. Also display the current percentage whenever the phone is removed or returned. Also add this percentage at the end of each log entry.
   5. **Cheating**: Monitor more frequently when present, to check for frequent changes in status that may indicate the phone is jiggling on the base because it is being used on the base. In addition to “present” and “away”, add the status of “jiggling” to the event log, with one line when the jiggling if first detected, and the following line will have when it is considered present or away again. Also, while in the jiggling state, display a flashing “?” on the LED display, and play a distinctive sound or pre-recorded message.
   6. **Speaker**: Connect the micro:bit to some speakers for louder volume. At start-up allow the user to choose between three pre-defined messages of your choice (eg “Bring it back soon”, “welcome back”), to be played when the phone is removed or returned. Use to text-to-speech functionality to play the selected message at the chosen time.
   7. **Monitor**: Connect to another micro:bit via radio. Write the code for both micro:bits so that the second one will display either “P” (for present) or “A” (for away), indicating whether the phone is currently on the base or not.
4. Prepare the following documentation and submit it on Schoolbox for marking:
   1. **User Manual**: Create a clear and concise user manual explaining how to use the micro:bit with your base, with instructions for all programmed features. (Maximum 2 x A4 pages)
   2. **Base Design and Evaluation**: Include a photo of the base you constructed, and explain the principles behind how it works. Also evaluate your design and implementation, explaining any weaknesses or strengths, and also explaining any possible improvements. (Maximum 400 words)
   3. **BBC micro:bit coding**: Provide a copy of your program code, including comments where explanation is helpful to the reader.
   4. **Coding Evaluation**: Write a self-evaluation of your code, identifying and explaining any short-comings, and evaluating the impact of these short-comings. (Maximum 400 words)
   5. **Video Evidence**: Create a short video clearly showing the base and micro:bit working together, and demonstrating that all features work as expected.
   6. **Sample File**: Include a copy of sample log.txt file, showing all the kinds of entries that can be recorded for the features you have coded.

**ASSIGNMENT MARKING SCHEME** (T Course – Total 30 Marks)

| **Item Assessed** | **A Grade**  (100% - 85% marks) | **B Grade**  (84% - 70% marks) | **C Grade**  (69% - 55% marks) | **D Grade**  (54% - 40% marks) | **E Grade**  (39% - 0% marks) | **Marks** |
| --- | --- | --- | --- | --- | --- | --- |
| User Manual | ▪ User manual explains how to use the micro:bit with base, with instructions for all programmed features  ▪ Consistently communicates clearly, concisely, and coherently with flair  ▪ Style and methods of communicating are always appropriate, and the result is highly effective and very engaging  ▪ Always uses technical terminology appropriately and correctly | ▪ User manual explains how to use the micro:bit with base, with instructions for most programmed features  ▪ Generally communicates clearly and logically with confidence  ▪ Style and methods of communicating are mostly appropriate, and the result is effective and somewhat engaging  ▪ Mostly uses technical terminology appropriately and correctly | ▪ User manual describes how to use the micro:bit with base, with instructions for most programmed features  ▪ Generally communicates clearly  ▪ Style and methods of communicating are generally appropriate, and in general can be readily understood.  ▪ Often uses technical terminology inappropriately and/or incorrectly | ▪ User manual describes how to use the micro:bit with base, with instructions for some programmed features  ▪ Often communicates without clarity  ▪ Style and methods of communicating demonstrate some attempt to be appropriate, and is often difficult to follow and understand.  ▪ Occasionally uses technical terminology appropriately and/or correctly | ▪ User manual identifies how to use the micro:bit with base, with instructions for some programmed features  ▪ Consistently communicates without clarity  ▪ Style and methods of communicating are not appropriate, and is mostly difficult to follow and understand.  ▪ Never uses technical terminology appropriately and/or correctly | /6 |
| Base Design Explanation and Evaluation | ▪ Constructed base is functional, able to detect whether the mobile phone is present or away  ▪ Explanation of base function is clear and easy to understand, concisely demonstrating a confident, accurate and in-depth understanding of the requirements and relevant theory and technologies  ▪ Critically analyses the base design evaluating its appropriateness and effectiveness and explaining any weaknesses or strengths, with any possible improvements  ▪ Always uses appropriate terminology with full understanding | ▪ Proposed solution meets all essential requirements  ▪ Explanation of base function is mostly clear, demonstrating an accurate understanding of the requirements and relevant theory and technologies  ▪ Analyses the base design explaining its appropriateness and effectiveness and explaining some possible weaknesses or strengths, describing possible improvements  ▪ Mostly uses appropriate terminology with full understanding | ▪ Proposed solution meets most essential requirements  ▪ Explanation of base function is generally clear, demonstrating a mostly accurate understanding of the requirements and relevant theory and technologies  ▪ Explains the appropriateness and effectiveness of the base design and describes some possible weaknesses or strengths, identifying possible improvements  ▪ Generally uses appropriate terminology with full understanding | ▪ Proposed solution meets some essential requirements  ▪ Explanation of base function demonstrates some accurate understanding of the requirements and relevant theory and technologies  ▪ Some description of the appropriateness and effectiveness of the base design, and some identifying of possible weakness or strengths  ▪ Sometimes uses appropriate terminology correctly | ▪ Proposed solution has little or no relation to essential requirements  ▪ Explanation of base function demonstrates some accurate understanding of relevant theory and technologies  ▪ Some identification of the appropriateness and effectiveness of the base design  ▪ Never uses appropriate terminology correctly | /4 |
| BBC micro:bit Common Coding | ▪ Implemented code is functional and high-quality, without errors and implementing all required features, demonstrating confident skills and understanding even with complex concepts  ▪ Comments are concise and clear, making the code more readable  ▪ Camel case is always used where appropriate | ▪ Implemented code is functional, mostly without errors and implementing most required features, demonstrating competent skills and understanding with some complex concepts  ▪ Comments are helpful, mostly making the code more readable | ▪ Successfully implemented most non-complex features, demonstrating competent skills and understanding with non-complex concepts, and able to significantly attempt more complex programming  ▪ Some helpful comments are included | ▪ Significant attempts to implement most non-complex features, demonstrating some skills and understanding with non-complex concepts  ▪ Some comments are included | ▪ Some attempt to implement non-complex features, demonstrating some skills with non-complex concepts  ▪ No comments are included | /7 |
| BBC micro:bit Additional Coding | ▪ Implemented code is functional and high-quality, without errors and implementing all required features, demonstrating confident skills and understanding even with complex concepts  ▪ Comments are concise and clear, making the code more readable | ▪ Implemented code is functional, mostly without errors and implementing most required features, demonstrating competent skills and understanding with some complex concepts  ▪ Comments are helpful, mostly making the code more readable | ▪ Successfully implemented most non-complex features, demonstrating competent skills and understanding with non-complex concepts, and able to significantly attempt more complex programming  ▪ Some helpful comments are included | ▪ Significant attempts to implement most non-complex features, demonstrating some skills and understanding with non-complex concepts  ▪ Some comments are included | ▪ Some attempt to implement non-complex features, demonstrating some skills with non-complex concepts  ▪ No comments are included | /7 |
| Coding Evaluation | ▪ Critically analyses the implemented code evaluating its appropriateness and effectiveness and explaining any weaknesses, evaluating the impact of any short-comings  ▪ Always uses appropriate terminology with full understanding | ▪ Analyses the implemented code evaluating its appropriateness and effectiveness and describing any weaknesses, explaining the impact of any short-comings  ▪ Mostly uses appropriate terminology with full understanding | ▪ Explains the implemented code evaluating its appropriateness and effectiveness and identifying any weaknesses, describing the impact of any short-comings  ▪ Generally uses appropriate terminology with full understanding | ▪ Describes the implemented code evaluating its appropriateness and effectiveness and identifying any weaknesses, identifying the impact of any short-comings  ▪ Sometimes uses appropriate terminology correctly | ▪ Identifies the implemented code with little or no reference to its appropriateness and effectiveness, and without identifying any weaknesses  ▪ Never uses appropriate terminology correctly | /4 |
| Video Evidence | ▪ Includes video evidence, demonstrating that all features work as expected |  | ▪ Includes video evidence, demonstrating some of the features working as expected. |  | ▪ Does not include video evidence. | /1 |
| Sample Log File | ▪ Includes sample log file, showing all the kinds of entries that can be recorded for the features coded, consistent with the implemented code |  | ▪ Includes sample log file, showing some of the kinds of entries that can be recorded for the features coded, consistent with the implemented code |  | ▪ Does not include sample log file, or sample log file has no relation to implemented code. | /1 |
| **TOTAL MARK** | | | | | | **/30** |