UG Exam Performance Feedback
First Year
2016/2017 Semester 1

COMP15111 Fundamentals of Computer Architecture

Javier Navaridas-Palma
Richard Neville

Comments

Feedback from Section A:

Very good overall performance for question A1 (basic stuff) showing that the students have been able to grasp the fundamentals of the course.

Some comments about A1 are:

a)Very good results in converting units between different bases.
A great proportion of students had mistakes in the simple arithmetic operations needed to do the conversion. These had been overseen iff the process to get the solution was correct. Some students took 2's compliment when translating from/to binary.
Although this was not the purpose of the exercise, such solutions were accepted. Most of the failures in this question came from performing octal division as if it was decimal (e.g. 13/2=5r1 in oct, not 6r1G.

b)Generally good performance understanding the different instructions and their impact on the architecture LDMFD is not restricted to be used for the stack, it can be used for many other purposes. Also one of the important points here was that the memory is accessed consecutively. Some students seemed to believe that LDR can not be used to access the stack, which is incorrect.

c)Very good results in understanding the need of registers and the overall architecture of a computer. Registers store data, not instructions.

Not many students attempted Question A2 but most students making a serious attempt to it got good marks. The results are acceptable overall, but the coding parts were the weakest.
Some comments from this part are:

a)Good understanding of how the stack operates and why is it needed. Some students forgot to mention that the stack is stored in memory and that it is necessary to safeguard state between method calls.

b)Memory addressing modes seem to have been grasped properly as well. The main comment here is that most student seem to believe that indirect addressing is restricted to be done through a register, but it could actually be done through main memory (although not in ARM). This has not been penalised. Some students mixed up pre- and post-indexing

c)Contrary to my expectations, this seems to be one of the most problematic questions, with some students trying to do somewhat unrelated (and more complicated) programs. The question was to calculate the address so to store the input character into its specific location within the very simple data structure (3-4 lines of code); some students did loops to get to the required address, or tried to print the board or some other unrelated stuff. I tried to mark accordingly based on
student's understanding of addressing and data structures.

d)This was the question that was attempted by the least number of students, possibly due to lack of time, but most of them showed a working understanding of using a stack. Some students used LDMFD/STMFD to access the stack, while using PUSH and POP is always preferable for code readability (this was not penalised). Some of the most common issues were to forget to remove the stack frame before returning from the method, and to forget to save registers in the stack(mostly LR). Also there were a few students that used caller saved, rather than callee saved as was explicitly asked for.

Section B

The feedback from RSN is provided in the attached report.

Section B1 & B2

1. Computer Architecture

General Feedback Comments

The following general comments are suggested to make your reflection and feedback more readable/succinct and viable for a more general succinct meditation on what you could do to enhance your learning and may be adapt your revision methodology.

First, it is important to reflect on the last lecture RN presented to the cohort; at the end of that lecture a comprehensive set of steps and guidance was presented for revision that students have advised me that they utilise. This was derived from methods students have utilised to revise over the years. The sentiments and guidance in this revision was sometimes given in their own words [the students]; and their own reflection on what worked best for different situations. But, may be it is worth noting that sometimes there can be discrepancies between the student's view of what mark they should have attained and what they actually were awarded. Reflecting on this issue may be it is worth noting and quoting the specific feedback from a student with respect to using the [suggested] self-test (or self-assessment) methodology, they said:

> "With regards to the self-assessment questions definitely allowed me to retain and recall large amounts of domain knowledge.
> It was especially useful in shorter questions and proved more beneficial than simply re-reading notes in a repetitive manner."

This is pertinent as without utilising a method like the self-assessment as well as undertaking a number of past exam papers one cannot self-access one's ability to pass the exam or access what mark one may obtain; and it is even more important when one goes into industry as without a good understanding of your own abilities how can you decide which courses to take, either those presented by the company that employs you or by external courses, and hence how can you evolve your personal skill set. You could also assesses where you feel you are in the **The Four stages of Learning (4SoL)): or Do you know what you know?**, information on this [4SoL] is on Blackboard 9.

Good companies will encourage you to undertake CPD. Continuing professional development (CPD) or Continuing professional education (CPE) is the means by which people maintain their knowledge and skills related to their professional lives.

One could say that evolving your revision and exam skill at University is, in fact, a form of CPD.

A final point, before getting into the detailed feedback for each question, is to reflect or ask yourself questions such as:

Did I undertaking past exam papers? [enough]

Did I undertaking a past exam paper –timed? [to get used to the time constraint of a real exam]

Did I develop a self-test (or self-assessment)? and finally

Did you utilise the methodologies presented in RN's exam revision lecture?

One of the recurring points was the advice on diagrams does not seem to be have been noted by many students; hence we repeat it hear: Remember, good – honours grade answer – in the exam – for a question – to maximise marks – should – or you should think of adding A DIAGRAM or a set of diagrams; hence a basic layout of a question answer may be:

1) Textual answer;
2) Diagram supporting answer [or code snippet]; &
3) Full explanation of diagram…

This sort of answer will [may] maximise your marks…

As per previous year's 79% of students answered section B2; which implies the majority (79%) of students answered questions from the second half of the lecture series.

Section B1

B1.a) Explain the term 'Garbage Collection' and explain why it is important to the implementation of Java. (4 marks)

B1.a) Application (Critique) (4 marks):

The following points should be covered to some degree in the answer:

i) Programs which use dynamic memory allocation may make use of the memory for a period but then, after a time, use it no longer. In these circumstances, a Garbage Collector tries to identify dynamically allocated memory which is no longer used and returns it to the memory allocation system.
Also, as we are continually creating new objects, we can run out of store. We would like to reclaim store; if it is no longer useful. An important part of the memory manager is a garbage collector: called when the heap is getting full; works out what is 'garbage;' e.g. un-referenced objects; no references [to it] exist …

ii) In Java, objects are continually allocated and discarded, so GC is very important. If this was not done by the GC the heap would fill up with un-referenced [unused] objects; this is important as the heap [nominally] has a set size (or limit on how much memory it can use) so the GC will stop the heap becoming full and remove unwanted [un-referenced] objects.

4 marks for an answer that depicts all the salient facts in a sensible way; good briefly described and expression is evaluated totally correct;
3 marks for correct answer but not detailed [enough];
2 mark for a right-lines approach;
1 mark for some basic understanding (or attempt).
Reference Learning Resources, Background Reading, and Lecture itself for detailed information; Lecture(s) No(s).: Lecture 18: Java Memory Usage.
TOTAL marks (4 marks) [4]

Marker's feedback
Pedagogic assessment [criterion]:
The question assesses Lecture 18: Java Memory Usage:.

Well done, most of you were able to state what "explain the term 'Garbage Collection' and explain why it is important to the implementation of Java" and you stated it concisely and explicitly.
The question's answer should clearly evidence knowledge of required salient facts relating to "explain the types of information the 'stack' holds."
In the answer [some of] the following terminology (keywords and naming conventions) should be utilised in context; for example: use dynamic memory, allocation, Garbage Collector, identify, allocated, no longer, used, returns, continually, creating, new, objects, run out of store, reclaim store, no longer useful, called, heap, getting full, un-referenced, objects,

no references, exist, continually allocated, discarded, fill up, [heap] set size, [GC] stop, full, remove unwanted; plus all those underlined in template answer in box above.
Main differentiation that must be clearly evidenced in your answer is that: "explain the term 'Garbage Collection'" and "explain why it is important to the implementation of Java," are stated in the template answer.
The question's answer should clearly evidence knowledge of required salient facts relating to "explain the term 'Garbage Collection'" and "explain why it is important to the implementation of Java"; this was not done explicitly in some of the answers given. If the answers did not detail what it is and its importance plainly as stated in the above (and in the Example answer); e.g. if answer does not give evidence of knowledge of required salient facts, full marks were not awarded.
This theory of "explain the term 'Garbage Collection'" and "explain why it is important to the implementation of Java" was covered in the lecture series in three different audio visual media: 1) the actual live lecture; 2) the audio recording of the live lecture; and 3) the real time video of the lecture.

B1.b) Two naming conventions are associated with garbage collection: 'live' and 'fragmentation.' Explain what each term means in the context of garbage collection. (2 marks)

B1.b) Bookwork (Explanation) (2 marks):

The following points should be covered to some degree in the answer:

Live: To discover if an object is garbage [or stale]; all references in the class variable and stack areas [or the heap] are followed to objects. If an object is referenced [in this context]; the object is marked as 'live;' e.g. objects in use and any references in those objects are followed to other objects are also marked 'live;' until no more unexplored references exist. If the object is not live it is a candidate for the garbage collector; and is marked as 'stale' and removed; a 'stale' object is an unreferenced object; which will be handled by the memory manager and reused.

Fragmentation: Every object on the heap is scanned; by the garbage collector. Any object not marked as garbage [marked as 'live'] will be removed by the garbage collector. Their store is 'returned' to the memory manager. But inevitably 'holes' form in memory where these objects were. After some time holes form across the entire memory this effect is usually known as 'fragmentation.'

2 marks *for an answer that depicts all the salient facts in a sensible way,*
1 1/2 marks *for correct answer but not detailed [enough],*
1 mark *for a right-lines approach,*
1/2 mark *for some basic understanding (or attempt).*

Reference Learning Resources, Background Reading, and Lecture itself for detailed information; Lecture(s) No(s).: Lecture 18: Java Memory Usage.
TOTAL marks (2 marks) [6]

Marker's feedback
Pedagogic assessment [criterion]:
The question assesses Lecture 18: Java Memory Usage.

Well done, most of you were able to state the difference between 'live' and 'fragmentation;' plus explain what each term means in the context of garbage collection.
The question's answer should clearly evidence knowledge of required salient facts relating to "the difference between 'live' and 'fragmentation;' plus explain what each term means in the context of garbage collection."
In the answer [some of] the following terminology (keywords and naming conventions) should be utilised in context; for example: discover, object, garbage, stale, all references, heap, live, followed to other objects, until no more unexplored, references exist, removed, unreferenced, Every object, scanned, by the garbage collector, returned, entire memory, fragmentation; plus all those underlined in template answer in box above.

Main differentiation that must be clearly evidenced in your answer is that [the]: difference between ‘live’ and ‘fragmentation;’ plus explain what each term means in the context of garbage collection, are stated in the template answer.
The question’s answer should clearly evidence knowledge of required salient facts relating to the “difference between ‘live’ and ‘fragmentation;’ plus explain what each term means in the context of garbage collection”; this was not done explicitly in some of the answers given. If the answers did not detail the mechanism plainly as stated in the above (and in the Example answer); e.g. if the answer does not give evidence of knowledge of required salient facts, full marks were not awarded.
This theory of “3 difference between ‘live’ and ‘fragmentation;’ plus explain what each term means in the context of garbage collection” was covered in the lecture series in three different audio visual media: 1) the actual live lecture; 2) the audio recording of the live lecture; and 3) the real time video of the lecture.

NOTE: It is important to note that when an object is tagged as ‘stale’ it is registered with memory manager which is scanning the memory; this then created a ‘hole’ in the memory termed a ‘fragment’; when the entire memory has been scanned these ‘holes’ or more correctly ‘fragments’ cause ‘fragmentation’ of the entire memory.
Hence, when answering the question, and in order to achieve full marks, one should associate ‘fragmentation’ with the entire memory.
It is also important when explain what each term means, in the context of garbage Collection, to state explicitly what ‘live’ and ‘fragmentation’ mean; e.g. explain how the memory manager: find, tags ‘live’ objects (which a referenced); and hence state the explicit fact that unreferenced objects are ‘stale’ objects; and the be explicit in your explanation of ‘live’ and ‘fragmentation’ as specified in the template answer; remembering to utilise the appropriate keywords in association with the appropriate explanation to explain how ‘live’ objects are processed and how ‘fragmentation’ occurs as a side effect of scanning the memory by the memory manager.

B1.c) Explain the principle of zero address instructions with the aid of an example of simple arithmetic expression evaluation. (4 marks)

B1.c) Bookwork and a little Application (4 marks):

The following points should be covered to some degree in the answer:

As the name implies, zero address instructions do not contain a memory address. Instead the operands are implicit, usually the top two locations of an evaluation stack. Also they should state: An instruction that contains no address fields; operand sources and destination are both implicit. It may for example enable stack processing: a zero-address instruction implies that the absolute address of the operand is held in a special register that is automatically incremented (or decremented) to point to the location of the top of the stack." "0-operand (zero address machines); so called stack machines: All arithmetic operations take place using the top one or two positions on the stack."

Example expression evaluation may contain some of the following zero-address instruction code; expect an expression evaluation e.g. (A+B)*(C-D) as an example. Should also mention that 1-address instructions are also usually needed to load and store to memory:

```
PUSH a  ; 1-address instruction; load from memory to stack
PUSH b ; 1-address instruction; load from memory to stack
ADD     ; 0-address instruction
PUSH c ; 1-address instruction; load from memory to stack
PUSH d ; 1-address instruction; load from memory to stack
SUB     ; 0-address instruction
MUL     ; 0-address instruction
POP  x   ; 1-address instruction; [store] from stack to memory
```

2 marks for an answer that depicts all the salient facts in a sensible way; all facts are correctly delineated and briefly described;
1 ½ marks for a right-lines approach;
1 mark for some basic understanding (or attempt).

Reference Learning Resources, Background Reading, and Lecture itself for detailed information; Lecture(s) No(s).: Lecture 17: Java bytecode.

TOTAL marks (4 marks) [10]

Marker's feedback
Pedagogic assessment [criterion]:
The question assesses Lecture 15: System Software.

Well done, most of you were able to state to "explain the principle of zero address instructions with the aid of an example of simple arithmetic expression evaluation".
The question's answer should clearly evidence knowledge of required salient facts relating to "explain[g] the principle of zero address instructions with the aid of an example of simple arithmetic expression evaluation."

In the answer [some of] the following terminology (keywords and naming conventions) should be utilised in context; for example: zero address, instructions, do not, contain, memory address, operands, implicit, top two locations, stack. Instruction, contains, no address fields, operand, sources, and destination, implicit, stack processing, zero-address, absolute address, special register, automatically incremented, decremented, stack, 0-operand zero address machines, stack machines, all arithmetic, top one or two, positions, stack; plus all those underlined in template answer in box above.
Main differentiation that must be clearly evidenced by "explaining the principle of zero address instructions with the aid of an example of simple arithmetic expression evaluation," are stated in the template answer.
The question's answer should clearly evidence knowledge of required salient facts relating to the issues aligned to "explaining the principle of zero address instructions with the aid of an example of simple arithmetic expression evaluation"; this was not done explicitly in some of the answers given.
If the answers did not detail the mechanism plainly as stated in the above (and in the Example answer); e.g. if the answer does not give evidence of knowledge of required salient facts, full marks were not awarded.
This theory of the System Kernel consists of basically five main functions was covered in the lecture series in three different audio visual media: 1) the actual live lecture; 2) the audio recording of the live lecture; and 3) the real time video of the lecture.

NOTE: one of the issues that a number of students had is they did not state that 1-address instructions are required to first move the data from the 'code variable area' onto the stack, e.g. PUSH a; and after the 0-address instruction another 1-address instruction 'POP c' is required to move the result from the stack back to the 'code variable area.' The most explicit means of signalling these differences, in the example code you write, would be to use the template answer style:

```
PUSH a  ; 1-address instruction; load from memory to stack
PUSH b ; 1-address instruction; load from memory to stack
ADD     ; 0-address instruction
PUSH c ; 1-address instruction; load from memory to stack
PUSH d ; 1-address instruction; load from memory to stack
SUB     ; 0-address instruction
MUL     ; 0-address instruction
POP  x   ; 1-address instruction; [store] from stack to memory
```

In your example arithmetic expression evaluation 0-address and 1-address instructions should be clearly delineated.

Which explicitly states which of the 1-address and 0-address instructions are actually "0-address instruction;" otherwise your answer does not explicitly state which is which or if you have not used the key words '1-address instruction' you would not have stated that only: ADD, SUB, & MUL… are 0-address instructions; which must be made explicit to gain full marks.

Section B2

B2.a) In the context of data exchange between CPU and peripherals. Differentiate between the two main data exchange protocols; polling and interrupts. (4 marks)

B2.b) Bookwork (2 marks)), Critique (2 marks) (4 marks):

The following points should be covered to some degree in the answer:

The two main data exchange protocols are:

Polling and interrupts; polling is initiated by the CPU, and interrupts are initiated by the peripherals.

1. Polling basically has two phases:

Polling phase 1: CPU polls the status register to check if a key has been pressed.
Polling phase 2: Then it reads the data register [in the peripheral] in order to transfer the data from the peripheral to the CPU. [Note this is in the context of a keyboard connected to a peripheral.]

2. Interrupts nominally [minimally] involve three steps:

Interrupts step 1: When the CPU starts the interrupt handler it first checks the status register. If the status [specified by a set bit] is incorrect the handler initiates an error handler.
Interrupts step 2: As long as the status facilitates a data transfer; the data register is then read; and the data is saved [in the appropriate location].
Interrupts step 3: Finally an acknowledgement is written back to the peripheral.
OR
In detail interrupts involve 5-steps which are**:**

1) **Stop** the program at the end of the current instruction;
2) Save (and maybe reset) the value of important registers;
3) Run the **interrupt handler** (method/code that deals with the peripheral);
4) Restore the value of the saved registers.
5) Finally, The program can now **continue** from its next instruction.

4 marks *for an answer that depicts all the salient facts in a sensible way,*
3 marks *for correct answer but not detailed [enough],*
2 mark *for a right-lines approach,*
1 mark *for some basic understanding (or attempt).*

Reference Learning Resources, Background Reading, and Lecture itself for detailed information; Lecture(s) No(s).: Lecture 14: Input/Output (1).
TOTAL marks (4 marks) [4]

Marker's feedback
Pedagogic assessment [criterion]:
The question assesses Lecture 15: System Software.

Well done, most of you were able to state what the "differentiate(s) between the two main data exchange protocols; polling and interrupts".
The question's answer should clearly evidence knowledge of required salient facts relating to what the "differentiate(s) between the two main data exchange protocols; polling and interrupts."
In the answer [some of] the following terminology (keywords and naming conventions) should be utilised in context; for example: Polling, CPU polls, status, register, check if, key,

pressed, reads, data, register, to transfer, data, from, peripheral, CPU. Interrupts, CPU starts, interrupt handler, first checks, status register, if, incorrect, handler, error handler, status, facilitates, data, register, read, data, saved, finally, acknowledgement, written back, peripheral; plus all those underlined in template answer in box above.

Main differentiation that must be clearly evidenced by "differentiate(s) between the two main data exchange protocols; polling and interrupts," are stated in the template answer.

The question's answer should clearly evidence knowledge of required salient facts relating to the issues aligned to "differentiate(s) between the two main data exchange protocols; polling and interrupts"; this was not done explicitly in some of the answers given.

If the answers did not detail the mechanism plainly as stated in the above (and in the Example answer); e.g. if the answer does not give evidence of knowledge of required salient facts, full marks were not awarded.

This theory of the "differentiate(s) between the two main data exchange protocols; polling and interrupts" was covered in the lecture series in three different audio visual media: 1) the actual live lecture; 2) the audio recording of the live lecture; and 3) the real time video of the lecture.

NOTE: One of the recurring issues, with those whom did not achieve full marks, was that they did NOT explicitly mention how the two differ by mentioning the sequence they both take; e.g. nominally two steps for polling; whereas interrupts have more steps, nominally three or more; reference template answer.

B2.b) The following array access code extract has some errors. Rewrite it correcting all the errors. There are six errors to be found in the code snippet below, figure B2.b. (3 marks)

```
1  LD     R0, i          ; i contains index
2  LDR    R , a          ; a contains base address
3  MOV    R2, #          ; because array of int
4  MUL    R3, R0  R2     ; i * 4 bytes
5  LDR    R ,[R1,R3      ; R4 = a[i];
```

Question figure B2.b. Code contains six errors.

B2.b) Application (4 marks):

The following points should be covered to some degree in the answer:

```
1  LDR[1]   R0, i          ; i contains index
2  LDR      R1[2], a       ; a contains base address
3  MOV      R2, #4[3]      ; because array of int
4  MUL      R3, R0,[4] R2  ; i * 4 bytes
5  LDR      R4[5],[R1,R3][6]  ; R4 = a[i];
```

3 marks for all six errors correctly found and corrected,
½ marks for each error corrected,
Reference Learning Resources, Background Reading, and Lecture itself for detailed information; Lectures 19: Input/Output (2).
TOTAL marks (3 marks) [7]

Marker's feedback
Pedagogic assessment [criterion]:
The question assesses Lecture 15: System Software.

Well done, most of you were able to "rewrite it [the code] correcting all the errors".
The question's answer should clearly evidence knowledge of required salient facts relating to what the "rewrite it [the code] correcting all the errors".
Main differentiation that must be clearly evidenced by "rewrite it [the code] correcting all the errors," are stated in the template answer.
The question's answer should clearly evidence knowledge of required salient facts relating to the issues aligned to "rewrite it [the code] correcting all the errors"; this was not done explicitly in some of the answers given.

If the answers did not detail the mechanism plainly as stated in the above (and in the Example answer); e.g. if the answer does not give evidence of knowledge of required salient facts, full marks were not awarded.
This theory aligned to rewrite it [the code] correcting all the errors was covered in: 1) the laboratories you under took; 2) in you example classes.

NOTE: the question does NOT ask you to re-write the code and optimize it; it states explicitly: “Rewrite it correcting all the errors;” it does not say change the instructions or rewrite and optimize the code; it ONLY required you to correct the code that was given; NOT change it in any way; those that did and did not correctly identify all the errors will not have been awarded full marks.

B2.c) What is the purpose of the Java Virtual Machine? (4 marks)

B2.c) Bookwork (2 marks)), Critique (2 marks) (4 marks):

The following points should be covered to some degree in the answer:

The Java compiler does not produce real machine instructions. It produces instead, bytecode, which is a stack based (zero address) instruction set. This has the advantages of portability and compactness. Bytecode is then executed by a program which: reads the code, performs the operations specified by them and maintains a store image of the program data. This is a virtual machine.

4 marks *for an answer that depicts all the salient facts in a sensible way,*
3 marks *for correct answer but not detailed [enough],*
2 mark *for a right-lines approach,*
1 mark *for some basic understanding (or attempt).*

Reference Learning Resources, Background Reading, and Lecture itself for detailed information; Lectures 17: Java bytecode.
TOTAL marks (4 marks) [11]

Marker's feedback
Pedagogic assessment [criterion]:
The question assesses Lecture 15: System Software.

Well done, most of you were able to state "What is the purpose of the Java Virtual Machine?".
The question's answer should clearly evidence knowledge of required salient facts relating to "What is the purpose of the Java Virtual Machine?".
In the answer [some of] the following terminology (keywords and naming conventions) should be utilised in context; for example: *Java, compiler, not produce, real, machine, instructions, bytecode, stack based, zero address, instruction, set, advantages, portability, compactness, executed, program, reads, code, performs, operations, maintains, store, image, program, data, virtual, machine*; plus all those underlined in template answer in box above.
Main differentiation that must be clearly evidenced by "what is the purpose of the Java Virtual Machine?," are stated in the template answer.
The question's answer should clearly evidence knowledge of required salient facts relating to the issues aligned to "what is the purpose of the Java Virtual Machine?"; this was not done explicitly in some of the answers given.
If the answers did not detail the mechanism plainly as stated in the above (and in the Example answer); e.g. if the answer does not give evidence of knowledge of required salient facts, full marks were not awarded.

This theory of “what is the purpose of the Java Virtual Machine?” was covered in the lecture series in three different audio visual media: 1) the actual live lecture; 2) the audio recording of the live lecture; and 3) the real time video of the lecture.

NOTE: to gain full marks the answer had to explicitly state the issues mentioned in the template answer. It is important to note the question does NOT ask about the dynamic compiler or the different compilers!
REMEMBER: the last laboratory covered the “JVM machine” and implementing it in ARM machine code; it has: “*a store image of the program data;*” an interpreter (implemented by a switch); and byte code was stored in memory in the ‘code’ area; so you have been given a detailed implementation of a JVM; and in the code you saw the JVM did not have a compiler (either just in time or dynamic); so when you compare the answer you gave remember what was the basic structure implemented in the JVM lab code.

B2.d) A ‘stack’ program is depicted in figure B2.d.1.

Instruction		
Address	Label	Mnemonic
000	`StackTop2`	`DEFW 0`
004	`StackStart2`	`DEFW 0`
008	`StackBottom2`	`DEFW 0`
00C	`StackProg2`	`MOV R0, #1`
010		`ADR SP, StackStart2`
014		`STR R0, [SP]`
018		`MOV R0, #2`
01C		`STR R0, [SP, #4]`
020		`SVC 2 ;(or SWI 2)`

Question figure B2.d.1. A ‘stack’ program; showing the address, Labels, & Mnemonics columns.

Describe in detail exactly what happens when the above ARM program is obeyed; using the table below, figure B2.d.2. In the table clearly describe the movement of information (both numbers and instructions) between memory, stack and the CPU in the comments column, and how the values in the registers R0, and R13 [SP] change, at each step.

Address/ label	Mnemonic	R0	R13 [SP]	**COMMENTS**
00C	`MOV R0, #1`	-	-	
010	`ADR SP, StackStart2`			
014	`STR R0, [SP]`	-	-	
018	`MOV R0, #2`	-	-	
01C	`STR R0, [SP, #4]`	-	-	
020	`SVC 2; (or SWI 2)`	-	-	

Question figure B2.d.2. A ‘stack’ program; showing the address, Mnemonics, R0, R13, & comments columns.

Assume that the program starts at memory location 0x00C; for this exam question.

In your answer please draw up a table, similar to the one above, which has the five columns for: addresses, mnemonics, registers and the comments in your answer book when you answer the question.

(9 marks)

B2.d) Application (example) (9 marks):

The following points should be covered to some degree in the answer:

Instruction		Registers		Comments
Add/ label	Mnemonic	R0	R13 [SP]	
00C	MOV R0,#1	**0x0000,0001**	-	Load R0 with data to be [pushed] stored on the stack; load Ro with #1.
010	ADR SP,StackStart2	**0x0000,0001**	**0x0000,0004**	Load SP address; Set up the Address of the stack pointer [SP], (SP=R13).
014	STR R0,[SP]	**0x0000,0001**	**0x0000,0004**	Store data (stored in R0) at the location pointed to by the SP, (SP=R13).
018	MOV R0,#2	**0x0000,0002**	**0x0000,0004**	Load R0 with data to be [pushed] stored on the stack; load R0 with #2.
01C	STR R0,[SP, #4]	**0x0000,0002**	**0x0000,0004**	Store data (stored in R0) at the location pointed to by the SP+4,(SP=R13)+4.
020	SWI 2	**0x0000,0002**	**0x0000,0004**	Stop [Halt] program.

caveat – note: Re-evolution of the model answer has all the above numbers in the next rows down. This is because in Reality the CPU [ARM] goes through the fetch-decode-execute cycle so the data is only valid after the instruction has run. So really the answers [numbers] should all be one row down. But if you state "ASSUMPTION: third row R0 and fourth row R13 – show data after instruction has executed." As the result you would see with Komodo would indeed only change the R0 & R13 registers after the instructions have executed; so in actuality would be one row down; e.g. 0x0000,0001 would appear in row 010 not 00c…

9 marks for all three [extra] columns [totally] correct,
4 marks for two of the columns correct,
2 marks for 1 one column, or some errors.
Reference Learning Resources, Background Reading, and Lecture itself for detailed information; Laboratory exercise 4, Lecture 9: Methods and Stacks & Lectures 18: Java Memory Usage.

TOTAL marks (9 marks) [20]

Marker's feedback
Pedagogic assessment [criterion]:
The question assesses Lecture 15: System Software.

Well done, most of you were able to state what the "describe in detail exactly what happens when the above ARM program - [by] draw up a table - and explicitly adding values in the R0 and R13 column, and comments explaining explicitly what each instruction does when it is executed, with respect to: R0 and R13, and the stack, and any other registers and associated addresses".
The question's answer should clearly evidence knowledge of required salient facts relating to what the "describe in detail exactly what happens when the above ARM program - [by] draw up a table - and explicitly adding values in the R0 and R13 column, and comments explaining

explicitly what each instruction does when it is executed, with respect to: R0 and R13, and the stack, and any other registers and associated addresses".
In the answer [some of] the following terminology (keywords and naming conventions) should be utilised in context; for example: load, data, push, stack, Address, pointer, stack pointer, Store, location, location pointed; plus all those underlined in template answer in box above.
Main differentiation that must be clearly evidenced when "describe[ing] in detail exactly what happens when the above ARM program - [by] draw up a table - and explicitly adding values in the R0 and R13 column, and comments explaining explicitly what each instruction does when it is executed, with respect to: R0 and R13, and the stack, and any other registers and associated addresses;" are explicitly stated in the template answer.
The question's answer should clearly evidence knowledge of required salient facts relating to the issues aligned to "describe[ing] in detail exactly what happens when the above ARM program - [by] draw up a table - and explicitly adding values in the R0 and R13 column, and comments explaining explicitly what each instruction does when it is executed, with respect to: R0 and R13, and the stack, and any other registers and associated addresses"; this was not done explicitly in some of the answers given.
If the answers did not detail the mechanism plainly as stated in the above (and in the Example answer); e.g. if the answer does not give evidence of knowledge of required salient facts, full marks were not awarded.
This theory [application of theory] "describe[d] in detail exactly what happens when the above ARM program - [by] draw up a table - and explicitly adding values in the R0 and R13 column, and comments explaining explicitly what each instruction does when it is executed, with respect to: R0 and R13, and the stack, and any other registers and associated addresses" was covered in the: laboratories, exercise classes, and lecture series in three different audio visual media: 1) the actual live lecture; 2) the audio recording of the live lecture; and 3) the real time video of the lecture.

NOTE: as it states, above in the question, "please draw up a table, similar to the one above, which has the five columns for: addresses, mnemonics, registers and the comments in your answer book when you answer the question;" some students did not:
1/ draw up a table;
2/ state the values in R0 and R13, after each instruction;
3/ Did not explicitly write the values in R0 and R13 for each instruction;
4/ Did not explicitly write the values in R0 and R13 for each instruction but used IMPLICIT dashes or hyphens, which implies nothing, and does not stare explicitly the values in the registers R0, and R13 [SP], at each step;
And could not be awarded maximum marks as they did not meet the requirements explicitly stated in the question.