CPSC 213 Topic Learning Goals

ALL Indeget ALL Indeget Second to a large is comparent with basic comparent with basin basin basic comparent with basin basic comparent with	Topic	ID	Learning Goals Students Can
Network No. Second sec			
Action Low 0 Image and a single Company and combines and a control combine prevaluation that Company and control control on the control contro control control control control contro control control control		A1	Describe a basic computer with basic components (ALU, Registers, Memory) and explain now instructions execute and data flows.
		D1	Trace execution of a simple Clargeron and translate to a set of machine lovel instructions to any late that Clargeron
Image: Proceedings of the second se			
		-	
		50	
Implementation Implementation 1 Species that metricines will addressing model empletering unitarial status at to be complete. 1 Complete addressing model empletering unitarial status at to be complete. 1 Complete addressing model empletering unitarial status at to be complete. 1 Complete addressing model empletering unitarial status at the data to the		B7	Recognize that subtracting a number from another involves taking the twos complement of the number and adding it. Be able to apply the principles of twos complement to be able to correct
			implement sign extension.*
G: Compare and certral the performance inpact of defension modes - specifically seals to datases the degram trate of this instructions are, emergy versus arginer access, and offer versus and specifically effect of defension of the performance. This specifically effect of defension of the performance. The performance of the performance of the performance. This specifically effect of defension of the performance. This specifically effect	ISA Design	C1	
Image: Index: doubles instruction set double. This handles discussion of infinitements, ortinguouhly, and orticlos, and performance. This should be down for pain of instructions up to the performance. This should be down for pain of instructions up to the performance. This should be down for pain of instructions up to the performance. This should be down for pain of instructions up to the performance. This should be down for pain of instructions up to the performance. This should be down for pain of instructions up to the performance. This should be down for pain of instructions up to the performance of the performance. This should be down for pain of instructions up to the performance. This should be down for pain of instructions up to the performance of the p		C2	Compare and contrast various addressing modes (e.g. the limitations of not supporting a particular mode in an instruction set, why dynamically generated addressing is necessary).
		C3	
Image of evolution process and the activity of the difference			
Image: construction Image: construction Image: construction 01 Develop in infiftment, baskeend output in dust winkings in bath as and C 01 Data is duffment, baskeend output in dust winkings in bath as and C 01 Data is build internation in status winkings in bath as and C 01 Data is build internation in status winkings in bath as and C 01 Data is build internation in status winkings in bath as and C 01 Data is build internation in status winkings in bath as and C 01 Data is build internation in status winkings in bath as and C 01 Data is build internation in status winking internation in the status internation in status winking internation in status winking internation in status winking internation in status winking internation in the status winking internation in the status winking internation in the status winking internation in status winking internatis winking internatis winking internatis		C4	
	Variables	D1	
31 Same for different Links of variables when information is statushy brown, and when information is dynamically brown. 120 Same for different Links of variables when information is statushy brown and when information is dynamically brown. 121 Same for different Links of variables when information is statushy brown and when information is dynamically brown (maintain). 121 Same for different Links of variables when information is statushy brown and when information is dynamically used results. 121 Same for different Links of variables when information is statushy brown and when informatis statushy brown and when infor	variables		
New of Conf. Else (seg tack d) gragen counter when cols using control from (sing) is excisted 1 Splin Ally control in (sing) is colded and tack in program the region to comple fryundicity when the subter states and the subter state in the subter state i			
2 Colubies provides based on the address of the program counter 3 Compace and control scored on the program counter 4 Compace and control scored on the program counter 4 Compace and control scored on the program counter 4 Compace and control scored on the program counter 4 Compace and control scored on the program control scored	Flow of Control	-	
Image:		F2	
14. Compare and nativation scenarios which regular static versus dynamic jump says where versus 14. Outcome in the scenarios which regular static versus dynamic jump says where versus 14. Outcome in the scenarios which regular static versus dynamic jump says where versus 14. Outcome in the scenarios which regular static versus dynamic jump says where versus 14. Outcome in the scenarios which regular static versus dynamic jump says where versus 14. Outcome in the scenarios where regular static versus dynamic jump says where versus 14. Outcome in the scenarios where regular static versus dynamic jump says where versus 14. Outcome in the scenarios where regular static versus dynamic jump says where versus 14. Outcome in the scenarios where regular static versus dynamic jump says where versus 14. Outcome in the scenario static versus dynamic jump says where versus dynamic versus dynamicyn versus dynamic versus dynamicyn versus dynamic versus dynamic		E3	
File Source C trans code samples with regure field enternal mining jung and use wersa E Source C trans code samples with regure field enternal backs to develop were use purp Eables to make source in transmission of the same of a programming language where preschere FIUMN was a staric jump Image: C pulp and transmission of the same of a programming language where preschere FIUMN was a staric jump Image: C pulp and transmission of the same of a programming language where preschere FIUMN was a staric jump Image: C pulp and transmission of the same of the same of a programming language where preschere FIUMN was a staric jump Image: C pulp and transmission of the same of the sa			
Image and proceeding return in close muta be dynamic - consider the case of a programming inguage whose procedors BTURNe was static jurg Image and the consequences to programming of elimitating ways allocade allocade inclusion dynamic return. Image and the consequences to programming of elimitating ways allocade allocade inclusion dynamic return. Image and the consequences to programming of elimitating ways allocade inclusion dynamic return. Image and the consequences to programming of elimitating ways allocade inclusion dynamic return. Image and the consequences to programming of elimitating ways allocade inclusion dynamic return. Image and the consequences to programming of elimitating ways allocade inclusion and the stack. Image and the consequences to programming of elimitating ways allocade inclusion and the stack. Image and the consequences allocade in the instruction and the stack and the dynamic return and decimating the format of the stack. Image and the consequences allocade in the instruction and the stack and the dynamic return and decimating in the consequences. Image and the stack and the consequences and the dynamic all return and decimating in the consequences. Image and the stack and the consequences and the dynamic allocade in the instruction and the stack in the consequences. Image and the stack and the consequences and the dynamic allocade in the instruction and the stack in the consequences. Image and the stack and the consequences and the dynamic and the stack and the stack in the instruction and the stack a		E5	
Draginal Image: Construction Image: Construction 17. rede Fight in the consequences to programming if lead variables and posed to just using the beap, including describing boot the stack is not required (e.g., you can just have a heap—and that have heads a stage in a stack is a stack i			
Process Explain the consequences to programming of eliminating dramatically allocated load variables and/or dynamic return. Process Explain the downlarge of using the stack for cold variables and/or stack is and required (e.g., you anjust have a heap - and that have and the have including describing how the stack is not required (e.g., you anjust have a heap - and that have a ward of the stack. Process Explain the downlarge of using the stack for cold variables and that have availed of the stack. Advanced students only. Process Explain the yaccentral unique relations and fifterent dyna use the long instead of the stack. Process Explain the yaccentral unique relations and different dyna use the long instead of the stack. Process Explain the yaccentral unique relations and different dyna use the long instead of the stack. Process Explain the yaccentral unique relations and different dyna use the long instead of the stack. Process Process Explain the yaccentral unique relations and the stack is and the different dyna use is a calles-save register and alter actively when to use a caller-save register by giving examples in machine code that be advant dyna thin dyna examples in machine code that the stack is and the different dyna use the cold process and different dyna use the stack is and the different dyna use is a different dyna use is and the different dyna use is a different dyna use is different dyna use is a different dyna use is a different dyna u			
F4 Explain the advantage of using the stack for local variables as opposed to jud using the hasp, including discribing how the stack is not required (e.g., vou can just have a hasp and that have the stack is a degram fraction?) F5 Show how procedure call implementation is different if you use the hasp including discribing how the stack. Advanced itudients only. F5 Show how procedure call implementation is different if you use the hasp including of the stack. Advanced itudients only. F7 Show how procedure call implementation is different if you use the hasp including of the stack. Advanced itudients only. F7 Show how procedure call implementation is different if you use the hasp including the stack is not registery. For example, describe how storing all values in the calle interview plant. F7 Show how dual dist dive characteristics contribute to how quality information case is registery and interview interview how quality information case is registery and interview in the stack is not registery. For example, describe how storing all values in the calle interview plant. F8 Explain with if 70 and DAA are and how hey guilter and are similar to each interview in the different information interview in the different information interview inter	-		
	Tradeoffs		
Fig. Slow how procedure call ingumentation is different 7 you use the heap instead of the stack. Fig. Slow how procedure call in governments a procedure call and return and describe the format of the stack. Fig. Slow the machine instruction necessary to implement a procedure call and return and describe the format of the stack. Fig. Slow the machine instruction necessary to implement a procedure call and return and describe the format of the stack. Fig. Slow the machine instruction necessary to implement approxemation of register usage (a, whit values to toru in register). For example, describe how strong all values in the called slow the instruction architecture alow. Fig. Slow the machine instruction necessary to implement approxemation of register usage (a, whit values to toru in register). For examples, in machine code that been fifter on each choice. Fig. Slow the machine instruction architecture is and calles cover register and alternatively when to use a caller-save register by gloing examples in machine code that been struction architecture is and calles cover register of the slow structure in the low guilely information con the settineed from data. Fig. Slow the Mice dual data everge in the structure in the system. Fig. Slow the machine in the system. Fig. Slow the Mice dual data everge in the system. Fig. Slow the system of the Sign data is and called cover in the system. Fig. Slow the Mice dual data everge in the System. Fig. Slow the Sign data everge in the Sign data		r4	
15 Understand advantage of multitationing accourse after a procedure returns and that this would require using the heap stands of the tack. Advantage of multitationing accourse after a procedure calling returns and the design tradeoffs of having it implements approaches the planning of regularity values to the intermstand of the tack. 16 Seption why de information coixs and the design tradeoffs of having it implements approaches the planning of regularity values to their intermstand. For example, design of the head the called completion coixs and the design tradeoffs of having it implements approaches. For example, design of the head the called completion coixs and the design tradeoffs of having it implements approaches. 10 Seption when the for and DNA and DNA and DNA they (Effer and an similar to each other implements approaches and the design tradeoffs of having it implements approaches. For applican head the design tradeoffs of having it implements approaches. 16 Seption when the design tradeoffs of having it implements approaches. For applican head the design tradeoffs of having it implements approaches. 17 Seption when the design tradeoffs of having it implements approaches. For applican head the design tradeoffs of having it implements approach on the design tradeoffs. 18 Seption head the design tradeoffs of having it implements approaches. For applican head the design tradeoffs of having it implements approaches. 19 Seption head the design tradeoffs of having it implements approaches. For applican head thead thead thead tradeoffs of hav		E5	
P7 Show the machine instruction necessary to implement a procedure call and return and discribe the forma of the task. P4 Explain why procedure-calling commonstrains dit dissign tradends of hubby implementation processary and the dissign and the instruction architecture alon. P57 Develop a heur sick that a complete cold use to determine when to use a caller-save register save register by gloing examples in machine code that P07 Develop a heur sick that a complete cold use to determine when to use a caller-save register by gloing examples in machine code that P07 Develop a heur sick that a complete cold use to determine when to use a caller-save register by gloing examples in machine code that P07 Develop a heur sick that a complete cold use to determine when to use a caller-save register by gloing examples in machine code that P07 Develop a heur sick that a complete cold use to determine when to use a caller-save register by gloing examples in machine code that P07 Develop a heur sick that a complete cold use to determine when to use a caller-save register by gloing examples in machine code that P07 Develop a heur sick that a complete cold use to determine when to use a caller-save register by gloing examples in machine code that P07 Develop a heur sick that a complete cold use to determine when the six that the that that the the six that the the six that the the six that a that the six that that the six that the six that a the six that a the si			
19 Copian how the indegendence of calles: and calles complicates the planning of register usage (e.g. what values to store in register). For example, describe how storing all values in the calle carey optimal. 10 Develop a heuristic that a complier could use to determine when to use a calles-save register and atternatively when to use a calles save register and atternatively when to use a calles save register and atternatively when to use a calles save and one calles atternatively when to use a calles save register and atternatively when to use a calles save register and atternatively when to use a calles save and one calles atternatively when to use a calles save and one calles atternatively when to use a calles save and one calles atternatively when to use a calles save and one calles atternatively when to use a calles save and one calles atternatively depending on the atture of the calles atternatively atternative atternatite atternat atternative atternat atternative atternative atternat			
Factor Construction External Construction Construction Construction Construction External Construction Cons		F8	
F10 [Device) a heuristic that a compiler could use to determine when to use a callee-save register and alternatively when to use a caller-save register by giving examples in machine code that bebeeft from acch holica.] Distance 61 Explain what FI0 and DMA are and how they differ and are similar to each other Processome 62 Explain what FI0 and DMA are and how they differ and are similar to each other Processome 62 Explain what FI0 and DMA are and how they differ and are similar to each other Processome 62 Explain how the control of the		F9	Explain how the independence of callers and callees complicates the planning of register usage (e.g. what values to store in register). For example, describe how storing all values in the caller is
Image: Contract Section 2016 Control 2016 Control 2016 Control 2016 Evelopies with 2016 Explain what 2016 and DMA are and how they differ and are similar to each other Devices with Explain what 2016 and DMA are and how they differ and are similar to each other Explain what 2016 and 2016 are and 2016 and 2016 are and 2016 and 2016 are			
Stema 61 Splain what hild and buck are and how they differ and are similar to each other Processent He Splain what hild wher characteristics contribute to beyong calch information can be retrieved from dial Hild Splain what hild wher characteristics contribute to beyong calch information can be retrieved from dial Hild Splain what hild wher characteristics contribute to beyong calch information can be retrieved from dial Hild Splain how tracteristics where the different crack sectori Hild Splain how tracteristics where the different crack sectori Hild Splain how tracteristics He Splain how tracteristics Hild Splain how tracteristics He Splain how traden how they diff haw schedune		F10	
Processent H1 Engine what disk dive characteristics contribute to how quickly information can be retrieved from div Processent Relative controls of the characteristics contribute to how quickly information can be retrieved from div Processent Replan and controls the tradeoffic dired, truck sector) Processent Replan how sectors are identified (head, truck sector) Processent Replan how sectors are identified (head, truck sector) Processent Replan how factors of the VMIX file system, basic building block and on disk data structures including blocks, index, and the vertice induce make of controls of the of tailure, depending on the status of the write in a file system. (this will kiely be going: www); Processent Resplan how factors the communication model for processor is a single machine (the procedure call model) versus networked communication (these differences) including goining familiarity with networking APMs; Describe how returned communication flows are any synchronas communication model in which synchronation meeds to be handed explicith? Processent Replan the synchronas synchronas communication model in which synchronation meeds to be handed explicith? Processent Replan the synchronas synchronas communication model in which synchronation meeds to be handed explicith? Processent Replan the synchronas synchronas communication model in which synchronation meeds to be handed explicith? Procostate and synchin synchr			
H2 Gelaultate working distance services H2 Calibulate working distance working distance working distance working distance working distance working distance distan		-	
H3 Explan how sectors are identified thead; track sectorf H4 Explan and compare the tradeoffs of dis scheduling agenthm make H5 Describe and draw pictures of the UNIX file system, basic building blocks and on disk data structures including blocks, indees, and file: H6 Explan and compare the tradeoffs of dis scheduling egenthm make H7 Describe how failure of the OS impacts various structures in the file system – at various points of time of failure, depending on the status of the write in a file system. This will likely be going: H7 Describe how reading each or the connection): H8 Compare and contrast the communication model for procedures on a single machine gelent and server getting connected, including gaining familianty with networking AMs: H8 Describe how reading each system system or substrating the server involving a client and server getting connected, including gaining familianty with networking AMs: H9 Describe how reading each system or each short with works schopping that stream into chunks, sending them independently, chunks can get-lost, and that reliability issues arise and muse be deal with: a protocol size space for each with a protocol size space. H7 Describe how reading each space for each with swith the index reading that processes access that the adversize getting. H7 Describe an antipe division for genoses space for each with a protocol size space for each with a protocol size space for each with swith readins for thom spin chronos	Files		
H4 Explain and congent the tradeoffs disk scheduling signifisms make. H4 Explain hand congent the tradeoffs disk scheduling signifisms make. H4 Apply hnowledge about disk performance characteristics to data layout on diak. H4 Explain hand congent the tradeoffs of singates various structures in the file system – at various points of time of failure, depending on the status of the write in a file system this will likely be going: awayh. Networking I Compare and contrast the communication model for procedures on a single machine (the procedure call model) versus networked communication (these differences include: make a concection): 9 Describe networked regregating the processor activative involves clopping that tream-intio chunks, sending them independently, chunks can get tost, and that reliability issues arise and muse be deal with: 9 Describe how sending signification processor activative involves clopping that tream-intio chunks, sending them independently, chunks can get tost, and that reliability issues arise and muse be deal with: 9 Processet 1 Protocor state, and isymmet for groups and tream into chunks, sending them independently, chunks can get tost, and that reliability issues arise and muse be deal with: 9 Inclain the dispit tradeoffs of why virtual addressing is needed and disrible and also the complexition (via base-bounds). 12 Explain that trends offs of any virtual addressis is needed and disrible and also the complexition.			
HS Describe and draw pictures of the UNIX file system, basic building blocks and a nikk data structures including blocks, index, and file; H2 Apply innovelage about disk performance characteristics to data layout on disk H3 Explain how failure of the QS impacts various structures in the file system, and values points of time of failure, depending on the status of the write in a file system; this will likely be going; awayj; Networking H1 Compare and contrast the communication model for procedures on a single machine (the procedure call model) versus networked communication (these differences include: make a connection); intended the system may anythenous communication model in which synchronization media to be handled explicitly. Performance that synchronization failows and synchroniza communication model in which synchronization media call bandland explicitly. Bescrite how networked communication model for procedure call model in which synchronization media call bandland explicitly. Bescrite how networked communication model for procedure call model in which synchronization media call bandland explicitly issues arise and muse be deal with the as protocol pitche sissues. Bescrite how networked communication model for procedure call and bas the complication and pitches is a protocol pitches space for acht process and that the than that ware loss the translation (via base-bounds). I2 Splain that during blocks band on an example of why we need to move from a synchronization active sissues. I3 Explain that processes are separate entities with ther anot addin			
H8 Explain how failure of the OE impacts various structures in the file system — at various points of time of failure; depending on the status of the write ine file system. (this will likely be going mwh). Networking 1 Compare and contrast the communication model for procedures on a single machine (the procedure call model) versus networked communication (these differences include: make a connection-include depilotity 12 Write a simple networked program (e.g. perhaps a very simple web serve: involving a client and server getting connected); including gaining familiarity with networking APIs: 13 Describe how networked program (e.g. perhaps a very simple web serve: involving a client and server getting connected); including gaining familiarity with networking APIs: 14 Describe how sending a stream of data across a network involves chooping that stream into chunks, sending them independently; chunks can get loss, and they into getting sing is needed and describe data across and the complication and performance implications. 15 Explain the design tradeoffs (via nalysis) is needed and describe and assis to exolation it's different and that this is an example of virtual memor 16 Trace though code with a producer/consumer relationship 17 Use synchronization primitives to enable implain groducer/consumer structures 18 Explain the denoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] 16 Trace though code			
wwy: wwy: Networking 1: Compare and contrast the communication model for procedures on a single machine (the procedure call model) versus networked communication (these differences include: make a communication relates the connection), transfer data, shut down the connection). 1: Write a simple networked program (e.g. perhaps a very simple web server involving a client and server getting connected), including gaining familiarity with networking APIs: 1: Describe how retworked communication follows an asynchronous communication model in which synchronization needs to be handled explicity. 1: Describe how retworked communication follows an asynchronous communication model in which synchronization needs to be handled explicity. 1: Describe how retworked dross pase (for each process and that that hardware does the translation (via base-bounds). 1: Explain that ther is a privite addross space for each process and that if two processes access the same address location it's different and that this is an example of virtual memory. 1: Describe anotivation for processes based on an example of why we need to move from asynchronous access to concurrent access with synchronization primitives: 1: Describe anotivation primitives to enable signaling in producer/consumer relationship. 1: Describe and work scenarios which require the use of concurrency via multi-threading. 1: Describe and work scenarios which require the used concurrency via wing the me		H7	Apply knowledge about disk performance characteristics to data layout on disk
Networking 4: Compare and contrast the communication model for procedures on a single machine (the procedure call model) versus networked communication (these differences include: make a comection, transfer data, shutd down the comection). 12 Write a simple networked program (e.g. perhaps a very simple web server involving a client and server getting connected), including gaining familiarity with networking APIs: 13 Describe how retworked communication follows an asynchronous communication model in which synchronization medels to be handled explicitly 14 Describe how retworked program (e.g. perhaps a very simple web server involving a client and synchronization medels to be handled explicitly 14 Describe how retworked communication follows an asynchronous communication that retain into chunks, sending them independently, chunks care get lost, and that reliability issues arise and music be dealt with: Describe a protocol plays in abstracting these issues: 16 IProtocol stack and layering (lessing is peeded and add desizable and also the complicating and performance implications. 13 Explain that there is a private address space of roach proceses a test and that if two proceses access the same address location it's different and that this is an example of virtual memor 14 Describe a motivation for proceses based on an example of virtual memor for a motivation primitives to enable implicit her address is an example of virtual memor 15 Describe a motivation primitives to enable imutual exclusion acces, e.g. using semaphores to control access to		H8	Explain how failure of the OS impacts various structures in the file system – at various points of time of failure, depending on the status of the write in a file system. (this will likely be going
accomparison comparison (e.g. performance) 12 Write a simple networked program (e.g. performance) are moving a client and server getting connected); including gating familiarity with networking APIs: 13 Describe how networked communication follows an asynchronous communication model in which synchronization needs to be handled explicitly 14 Describe how networked communication follows an asynchronous communication model in which synchronization needs to be handled explicitly 15 Protocol stack and layering (design, layers of abstracting ping that stream in the chains, sending them independently, chunks can get lost, and that that hardware does the translation (via base-bounds). 16 Explain that there is a private address space of needs process and that that hardware does the translation (via base-bounds). 17 Explain that there is a private address space of needs process and that that hardware does the translation (via base-bounds). 18 Explain that there is a private address space and that if two processes a ccess to concurrent access with synchronization primitive. 17 Describe an obtacation of processes bare of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] 16 Trace though code with a producer/consumer relationship. 17 Use synchronization primitives to enable singling in producer/consumer structures 18 Use synchronization primitives to enable			
Image: Processes Processes are separate entities with earmonication follows an asynchronication model in which synchronization needs to be handled explicitly Image: Processes Image: Processes Processes Image: Processes Processes Processes Image: Processes Processes Image: Processes	Networking	l1	
 Besribe how networked communication follows an synchronous communication model in which synchronization needs to be handled explicitly Besribe how sending a stream of data across a network involves chopping that stream into chunks, sending them independently, chunks can get lost, and that reliability issues arise and mus be dealt with- Describe the role that a protocol plays in abstracting these issues: Processes Explain there is a private address space for each process and that that hardware does the translation (via base-bounds). Explain there is a private address space of cach process and that that hardware does the translation (via base-bounds). Explain the design tradeoffs of why virtual addressing is needed and desirable and also the complicating and performance implications. Explain the design tradeoffs of why virtual addressing is needed and desirable and also the complicating and performance implications. Explain the design tradeoffs of why virtual addressing is needed and desirable and also the complicating and performance implications. Explain the design tradeoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] Trace though code with a produce/consumer relationship. Use synchronization primitives to enable mutal exclusion access, e.g. using semaphores to control access to a shared array. Use synchronization primitives to enable signaling in produce/consumer structures Compare and contrast the synchronization features that students already know from java with the variations available in C and Unix Compare and contrast the synchronization features that students already know from java with the variations available in C and Unix Use C syntax for poprieses of exerestructures of locking. Can explain the standerd dining philos		L2	
Image: Proceeding - attemp - of data carces a network involves chopping that stream into chunks; sending them independently, chunks can get-lost, and that reliability issues arise and mus be dealt with. Describe the role that a protocol plays in abstraction, be dealt with. Describe the role that a protocol plays in abstracting these issues: If (Protocol tack and layering (design, layers of obstraction) on covered in 231 Processes I Explain that there is a private address pis to reded and desished and also the complicating and performance implications. I Explain that processes are separate entities with their own address space and that if two processes access the same address location it's different and that this is an example of virtual memory I/4 Describe at a basic level the tradeoffs (via analysis with examples of round rolin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] I/6 Trace though code with a producer/consume relationslip. I/7 Use synchronization primitives to enable ginaling in producer/consume relationslip. I/10 Use synchronization primitives to enable ginaling in produce/consume relationslip. I/10 Use synchronization primitives to enable ginaling in produce/consume relationslip. I/10 Use synchronization primitives to enable ginaling in produce/consume structures. I/11 Use synchronization primitives to enable ginaling in produce/consumestructures. I/12 <td></td> <td></td>			
be dealt with. Describe the role that a protocol plays in abstracting these issues. IDE Protocol stack and layering (design, layers of bastraction)not covered in 213 Processes II Explain that there is a private address space for each process and that that hardware does the translation (via base-bounds). IDE Explain that dress in tradeoffs of why virtual addressing is needed and desirable and also the complicating and performance implications. IDE Explain that processes are separate entities with their own address space and that if two processes access the same address location it's different and that this is an example of virtual memory IDE Describe a notivation for processes based on an example of why we need to move from asynchronous access to concurrent access with synchronization primitives. IDE Use synchronization primitives to enable signaling in producer/consumer structures. IDE Use synchronization primitives to enable signaling in producer/consumer structures. IDE Explain how threads and processes offer specifically with regards to shared memory. IDE Compare and contrast the synchronization from features that students already know from Java with the variations available in C and Unix. IDE Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix. IDE Compare and contrast the synchronization features that students already		13 14	
If IProtocol-stack and layering (design, layers of abstraction)not covered in 213 Processes I Explain that there is a private address space for each process and that that hardware does the translation (via base-bounds). Explain that processes are separate entities with their own address space and that if two processes access the same address location it's different and that this is an example of virtual memor J4 Describe a motivation for processes based on an example of why we need to move from asynchronous access to concurrent access with synchronization primitive: J5 [Describe at a basic level the tradeoffs (wa analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] J6 Trace though code with a producer/consumer relations/lp. J7 Use synchronization primitives to enable mutual exclusion access, e.g. using semaphores to control access to a shared array. J8 Use synchronization primitives to enable inplain in producer/consumer structures J8 [Use synchronization primitives to enable inplain in producer/consumer relations/no thor adverting and contrast the synchronization features that students already know from Java with the variations available in C and Unix J10 Describe real world scenarios which require the use of cocking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with liferent granularities of locking. Can explain the standard dining philosophers' problem. Priority inve			
Processes 11 Explain that there is a private address space for each process and that that hardware does the translation (via base-bounds). 12 Explain that there is a private address space for each process and that that hardware does the translation (via base-bounds). 12 Explain that there is a private address space for each process access the complicating and performance implications. 14 Describe a motivation for processes based on an example of why we need to move from asynchronous access to concurrent access with synchronization primitive: 15 [Describe at a basic level the tradeoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] 16 Trace though code with a producer/consumer relationship 17 Use synchronization primitives to enable signaling in producer/consumer structures 18 Use synchronization primitives to enable signaling in producer/consumer structures 19 Explain how threads and processes converts asynchrony into concurrency by using synchronization primitives. Interrupts. 111 [Would like: Explain how processes converts asynchrony thro diverts at students already know from Java with the variations available in C and Unix 113 Compare and contrast the synchronization fratures that to diverte the possibility of deadlock. Give an example of acode that has the possibility of deadlock. Give an example of acode that has the possibility of deadlocking and a different example of with		15	
13 Explain that processes are separate entities with their own address space and that if two processes access the same address location it's different and that this is an example of virtual memory 14 Describe at abasic level the tradeoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] 16 Trace though code with a producer/consumer relationship. 17 Use synchronization primitives to enable mutual exclusion access, e.g. using semaphores to control access to a shared array. 18 Use synchronization primitives to enable mutual exclusion access, e.g. using semaphores to control access to a shared array. 19 Explain how threads and processes differ specifically with regards to shared memory 10 Describe real world scenarios which require the use of concurrency via multi-threading 111 [Would like: Explain how processes converts asynchrony into concurrency via multi-threading 121 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix 133 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with life? 134 [Explain how more addifferences between C structs and Java arrays are dynamic). I3 134 Use C syntax for pointers and compare that to re	Processes	J1	
Image: A standard of the standard the standard the standard the standard the standard the standard the		J2	Explain the design tradeoffs of why virtual addressing is needed and desirable and also the complicating and performance implications.
J5 [Describe at a basic level the tradeoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] J6 Trace though code with a producer/consumer relationship. J7 Use synchronization primitives to enable signaling in producer/consumer structures J9 Explain how threads and processes differ specifically with regards to shared memory J10 Describe real world scenarios which require the use of concurrency by using synchronization primitives to anable signaling in producer/consumer structures J12 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix J13 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix J14 [Kvold like: Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. C an explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with trip J3va and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types K3 Use C syntax for pointers and dompare that to reference variable use in Java.	l	13	Explain that processes are separate entities with their own address space and that if two processes access the same address location it's different and that this is an example of virtual memory.
J5 [Describe at a basic level the tradeoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.] J6 Trace though code with a producer/consumer relationship. J7 Use synchronization primitives to enable signaling in producer/consumer structures J9 Explain how threads and processes differ specifically with regards to shared memory J10 Describe real world scenarios which require the use of concurrency by using synchronization primitives to anable signaling in producer/consumer structures J12 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix J13 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix J14 [Kvold like: Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. C an explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with trip J3va and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types K3 Use C syntax for pointers and dompare that to reference variable use in Java.			
16 Trace though code with a producer/consumer relationship. 17 Use synchronization primitives to enable mutual exclusion access, e.g. using semaphores to control access to a shared array. 18 Use synchronization primitives to enable signaling in producer/consumer structures 19 Explain how threads and processes differ specifically with regards to shared memory 100 Describe real world scenarios which require the use of concurrency ia multi-threading 111 [Would like: Explain how processes converts asynchrony into concurrency by using synchronization primitives. Interrupts. 120 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix 131 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C 14 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types K4 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). 13 K4 Describe how memory reclamation is differences between C structs and Java objects	ļ		
J7 Use synchronization primitives to enable mutual exclusion access, e.g. using semaphores to control access to a shared array. J8 Use synchronization primitives to enable signaling in producer/consumer structures J9 Explain how threads and processes differ specifically with regards to shared memory J10 Describe real world scenarios which require the use of concurrency by using synchronization primitives. Interrupts. J11 [Would like: Explain how processes converts asynchrony into concurrency by using synchronization primitives. Interrupts. J12 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix J13 Compare and contrast the threading features that students already know from Java with the variations available in C and Unix J14 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C K1 Write C code equivalent to known Java Code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java alava arrays are dynamic). <t< td=""><td></td><td>J5</td><td>[Describe at a basic level the tradeoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.]</td></t<>		J5	[Describe at a basic level the tradeoffs (via analysis with examples of round robin, the role of the kernel clock, pre-emption, interrupts) available for scheduling of processes.]
 Use synchronization primitives to enable signaling in producer/consumer structures Explain how threads and processes differ specifically with regards to shared memory Describe real world scenarios which require the use of concurrency via multi-threading Would like: Explain how processes converts asynchrony into concurrency by using synchronization primitives. Interrupts. Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix Compare and contrast the threading features that students already know from Java with the variations available in C and Unix Compare and contrast the threading features that students already know from Java with the variations available in C and Unix Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it? Java and C Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types Compare and compare that to reference variable use in Java. Bescribe the similarities and differences between C structs and Java adva advite features are addressed in assembly code Describe how memory reclamation is different and be able to write C programs that use memory reclamation Describe how memory reclamation is different and be able to write C programs that use memory reclamation Creat a jump table to implement a C switch statement. Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leak	[J6	
19 Explain how threads and processes differ specifically with regards to shared memory 10 Describe real world scenarios which require the use of concurrency by using synchronization primitives. Interrupts. 111 [Would like: Explain how processes converts asynchrony into concurrency by using synchronization primitives. Interrupts. 112 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix 113 Compare and contrast the threading features that students already know from Java with the variations available in C and Unix 114 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it? Java and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). K4 Describe that similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. Mid		J7	
J10 Describe real world scenarios which require the use of concurrency via multi-threading J11 [Would like: Explain how processes converts asynchrony into concurrency by using synchronization primitives. Interrupts. J12 Compare and contrast the threading features that students already know from Java with the variations available in C and Unix J13 Compare and contrast the threading features that students already know from Java with the variations available in C and Unix J14 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe the similarities and different and be able	ļ		
J11 [Would like: Explain how processes converts asynchrony into concurrency by using synchronization primitives. Interrupts. J12 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix J13 Compare and contrast the threading features that students already know from Java with the variations available in C and Unix J14 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types compare and contrast for pointers and compare that to reference variable use in Java. K2 Describe how arrays are different in C and Java (C arrays are static and Java objects and specifically how their features are addressed in assembly code K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and different and be able to write C programs that use memory reclamation K5 Do pointer arithmetic in C. K6 Describe how memory reclamation is different and be able to write C programs that use memory reclamat			
J12 Compare and contrast the synchronization features that students already know from Java with the variations available in C and Unix J13 Compare and contrast the threading features that students already know from Java with the variations available in C and Unix J14 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C compare and contrast the similarities of C that is basically the same in both languages – primarily the imperative structures and primitive types Compare and contrast the similarities and different in C and Java (C arrays are static and Java arrays are dynamic). K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and different granularities on Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K1 Describe wy Java's polymorphism required indirect jumps and discuss the pe			
113 Compare and contrast the threading features that students already know from Java with the variations available in C and Unix 114 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code K6 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two probl			
J14 [Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that			
 with live-locking. Explain the tradeoffs associated with different granularities of locking. Can explain the standard dining philosophers' problem. Priority inversion and techniques for dealing with it?] Java and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types Comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how arrays accillectors only solve one of thes two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 			Explain how when a program has more than one lock, that it introduces the possibility of deadlock. Give an example of a code that has the possibility of deadlocking and a different example
Java and C K1 Write C code equivalent to known Java code (for the subset of C that is basically the same in both languages – primarily the imperative structures and primitive types comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how grabage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that			
comparable K2 Describe how arrays are different in C and Java (C arrays are static and Java arrays are dynamic). K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that			
 K3 Use C syntax for pointers and compare that to reference variable use in Java. K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how memory reclamation is different and be able to write C programs that use memory reclamation K8 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 			
 K4 Describe the similarities and differences between C structs and Java objects and specifically how their features are addressed in assembly code K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how memory reclamation is different and be able to write C programs that use memory reclamation K8 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 	comparable		
 K5 Do pointer arithmetic in C. K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how memory reclamation is different and be able to write C programs that use memory reclamation K8 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 			
 K6 Describe that dynamic memory allocation is the same in C and Java but that type safety is different K7 Describe how memory reclamation is different and be able to write C programs that use memory reclamation K8 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 	ļ		
 K7 Describe how memory reclamation is different and be able to write C programs that use memory reclamation K8 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 			
 K8 Describe how garbage collectors only solve one of these two memory problems: dangling pointers and memory leaks (including being able to describe these two problems and give code examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 	-		
examples which would create them). K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that			
 K9 Create a jump table to implement a C switch statement. K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that 			
K10 Describe why Java's polymorphism required indirect jumps and discuss the performance implications of that		К9	
		-	