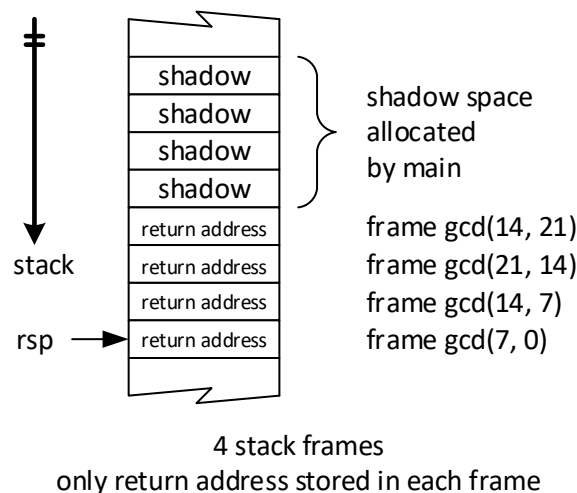


CSU34021 Tutorial 2 Notes

- (i) In general, your assembly language program needs to be better commented.
- (ii) The code for `p()` is not quite straight forward as it might at first seem. First shadow space needs to be allocated by `p()` when it calls `min()`. Must assume that `min()` needs shadow space. `P()` contains two calls to `min()`. Need only allocate shadow space once and “reuse” for each call to `min()`. Parameters `k` and `l` are passed to `p()` in `r8` and `r9`. Must ensure that these parameters survive the first call to `min()` which is quite entitled to overwrite `r8` and `r9` as they are volatile registers according to the x64 procedure calling convention. Example code saves `r8` and `r9` in `p`’s shadow space.
- (iii) There is no need to allocate shadow space for the recursive calls to `gcd`. Treat `gcd` as a leaf function.
- (iii) `a%b` should be calculated using `idiv`. `idiv` uses signed arithmetic whilst `div` uses unsigned arithmetic. `idiv` divides `rdx:rax` (128 bits) by the instruction operand (64 bits). The quotient is returned in `rax` and the remainder in `rdx`. `rdx` should be initialised using `cqo` as it sign extends `rax` across `rdx`. Zeroing `rdx` is not the same, although it will work with the examples given (need better test cases).
- (iv) Some students still had trouble with global variable `g` which needs to be allocated in `t2.asm` and its “interface” specified in `t2.h` (see sample answer).
- (v) Layout of stack frames must match submitted code.



- (vi) Some students did not manage to get `printf()` working in `q()`. Study the example solution. NB: solution assumes that `printf()` could overwrite any parameters passed to it on the stack (can’t find any official documentation on this issue).
- (vii) A simple function `qns()` is provided which generates an exception when called as it does not allocate shadow space before calling `printf()`.

- (iv) Some students did not manage to get `printf()` working in `q()`. Study the example solution. NB: solution assumes that `printf()` could overwrite any parameters passed to it on the stack (can't find any official documentation on this issue).
- (v) A simple function `qns()` is provided which that generates an exception when called as it does not allocate shadow space before calling `printf()`.

Additional comments from Harshvardhan Pandit:

- (vi) Shadow spaces are allocated on the stack as their purpose is to provide a temporary or 'scratch space' to write data regarding during the current call which is discarded or overwritten upon return.
- (vii) Blackboard allows submission of multiple files. Zipping them together only makes them more difficult to mark as it requires the file to be downloaded, unzipped and opened.
- (viii) Submission of entire Visual Studio projects is not expected or required and is discouraged since it makes it more time consuming to mark.
- (ix) When the question asks for code files (`t.asm` and `t.h`) – it asks for text files containing the respective code. Providing the code in DOCX and PDF files is not expected. It is okay if the code files are accompanied by 'documentation'.
- (x) Outputs are expected to provide answer to the solution in addition to 'proof' - for example, when Q4 asks about executing `qns()` without shadow space, merely providing a screenshot of an exception is not sufficient to answer the question. It must be accompanied with your analysis of why the exception occurred.
- (xi) Code outputs should be complete in a solution, for example, when asking for output of an execution, the window should show the entire output (including console window – not entire monitor/screen). There is no need for clipping the output or reducing it in any form.
- (xii) Code should be commented and should have a readable layout.