Pre-Lecture 5

Due Sep 4 at 9am **Points** 12 **Questions** 8 **Available** until Sep 4 at 9am
Time Limit None **Allowed Attempts** 2

Instructions

Take this quiz ***after you have watched the required videos and/or read the associated sections of the textbook***. See **Lecture 5: Representing procedures**.

You may attempt this quiz twice. Incorrect responses are marked after each attempt. Correct answers are revealed at the start of class for this lecture.

Carefully note the deadline for responses. Submissions are not accepted after the deadline, and there is no grace period.

This quiz was locked Sep 4 at 9am.

Attempt History

	Attempt	Time	Score
LATEST	**Attempt 1**	4,779 minutes	12 out of 12

Score for this attempt: **12** out of 12
Submitted Sep 3 at 10:19pm
This attempt took 4,779 minutes.

Question 1 — 2 / 2 pts

When we draw pictures of the program stack, we draw the bottom of the stack ***above*** the top of the stack. Push operations cause the stack to grow toward lower / smaller memory addresses, while pop operations cause the stack to shrink toward higher / larger memory addresses.

Answer 1:

Correct!

bottom

Answer 2:

Correct! top

Answer 3:

Correct! lower / smaller

Answer 4:

Correct! higher / larger

Question 2

1 / 1 pts

Consider the following C program:

```
01  int f(int n) {
02    if(n == 1)
03      return 1;   // this line
04    return n + f(n-1);
05  }
06
07  int main() {
08    int result = f(5);
09    printf("The factorial of 5 is %d\n", result);
10  }
```

Suppose that the size of each stack frame for procedure f is 32 bytes. Also suppose that immediately before the call to f in main, %rsp contains 0x7fffffffdf70. Provide the contents of %rsp immediately before the commented line in f is executed.

%rsp contains 0x 7fffffffded0 .

(Assume a direct translation to x86 with no optimizations.)

Answer 1:

Correct! 7fffffffded0

orrect Answer 7FFFFFFFDED0

Question 3

2 / 2 pts

Suppose that we have disassembled a program and see these two lines at the site of a procedure call:

```
...
0x4004f6: callq 0x40050c
0x4004fb: mov $0x0, %eax
...
```

Execution of the callq instruction causes address 0x 4004fb to be pushed onto the stack, and address 0x 40050c to be placed in %rip.

Answer 1:

Correct!

4004fb

Answer 2:

Correct!

40050c

Question 4

1 / 1 pts

The value of which argument is placed into register %rax to prepare for a procedure call?

- 1st argument
- 2nd argument
- 3rd argument
- 4th argument
- 5th argument

- 6th argument
- Correct! none of the above

Question 5

1 / 1 pts

Where is the value of a potential 7th argument placed to prepare for a procedure call?

- in register %rsi
- in register %r9
- Correct! in memory at the address (%rsp)
- in memory at the address 8(%rsp)
- in memory at the address -8(%rsp)
- none of the above

Question 6

2 / 2 pts

Consider the following C program:

```
int p(int a, int b) {
  return q(b, a) + 3a - 2b;
}

int q(int x, int y) {
  return x + y;
}

int main() {
  int result = p(1, 2);
  printf("The answer is %d\n", result);
}
```

Notice that both p and q use the first two argument registers.

Who is responsible for saving the values of p's two arguments, to ensure that they are available for use after the call to q? p as the caller of q

Which x86 operation is used to save the contents of a register? push

Which x86 operation is used to restore the contents of a register? pop

Answer 1:

Correct!

p as the caller of q

Answer 2:

Correct!

push

Answer 3:

Correct!

pop

Question 7

1 / 1 pts

What is the exact compiler flag one can use with gcc so that helpful information (e.g., function names, line numbers) can be seen when debugging with gdb?

Correct!

-g

orrect Answers

-g

g

Question 8

2 / 2 pts

Match the x86 instruction to where it belongs in a procedure's prolog or epilog.

Correct! **1st prolog instruction** pushq %rbp

Correct! **2nd prolog instruction** movq %rsp, %rbp

Correct! **1st epilog instruction** popq %rbp

Correct! **2nd epilog instruction** retq

Other Incorrect Match Options:

- movq %rbp, %rsp
- pushq %rsp
- popq %rsp

Quiz Score: **12** out of 12