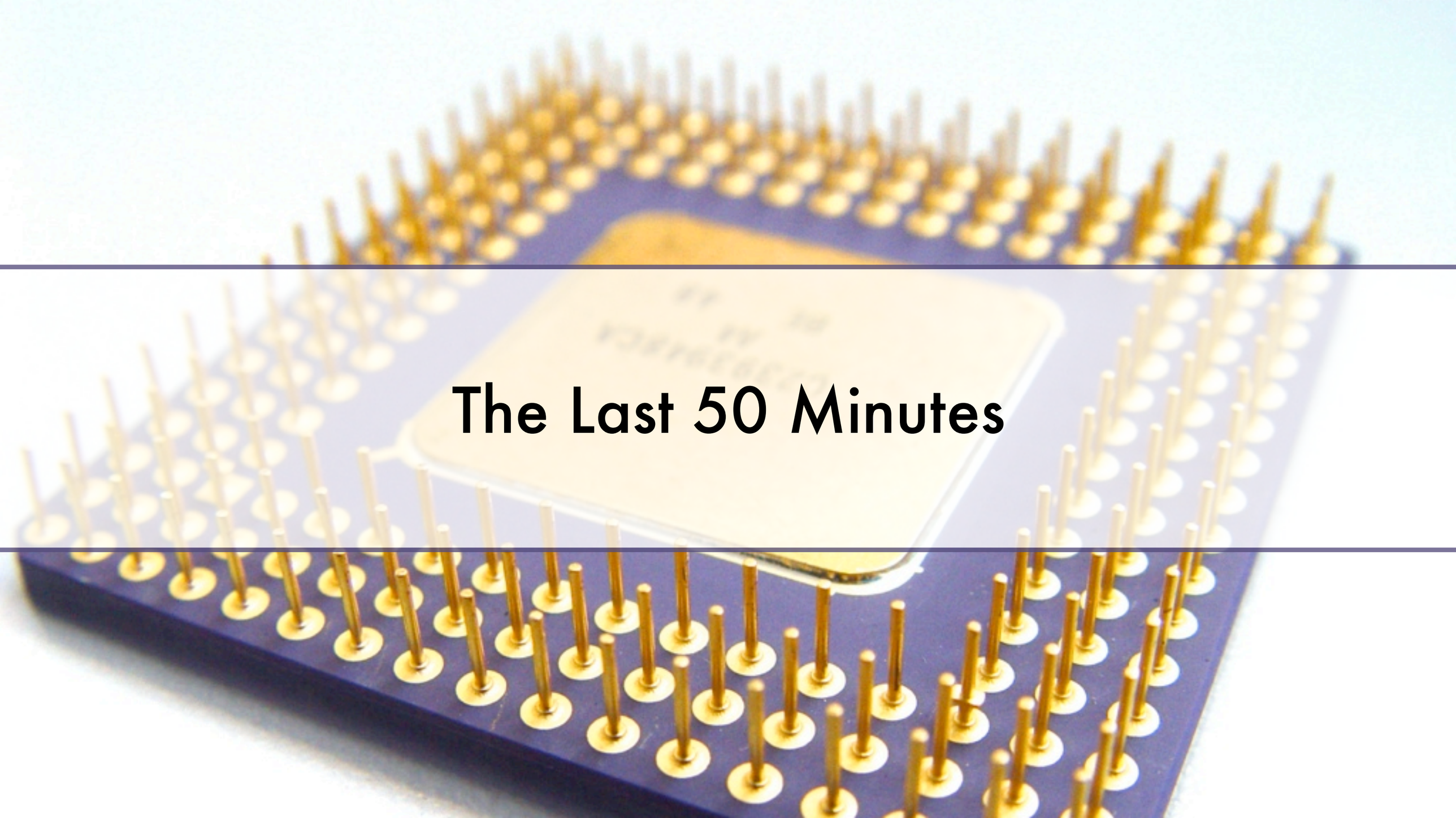
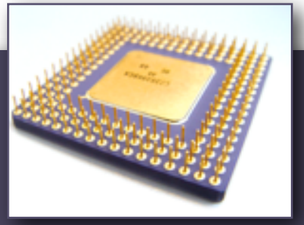
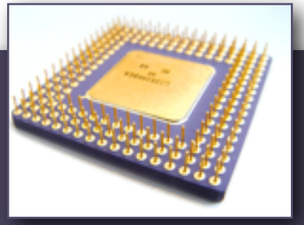


The Last 50 Minutes

Administrivia: Final Grades

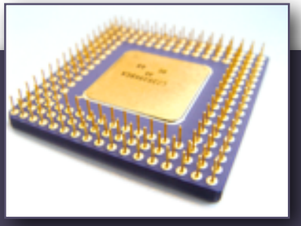


Administrivia: Final Grades



- All grades are posted *except* Homework 6, Lab 6, and a few attendance points

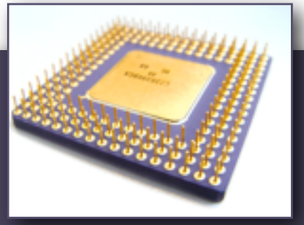
Administrivia: Final Grades



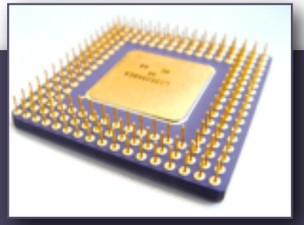
- All grades are posted *except* Homework 6, Lab 6, and a few attendance points

- **Final letter grade cutoffs are 90.0/80.0/70.0/60.0 strict**
  - When Harika posts HW6 grades, e-mail her *immediately* if there's a problem
  - You're welcome to come by my office to review your final exam (I keep the exams)
  - No "fishing for points" – too late for grade changes for Homework 1–5 and Exams 1–2
  - No special favors, ever

Administrivia: Final Exam

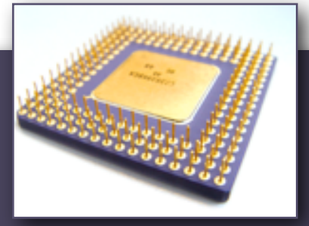


Administrivia: Final Exam



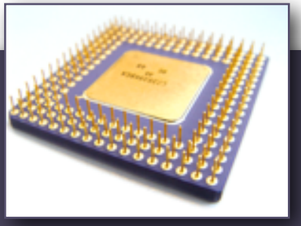
- **Friday, December 12, 12:00–2:30 p.m.**

Administrivia: Final Exam



- **Friday, December 12, 12:00–2:30 p.m.**
- **Allowed one double-sided 8½×11” cheat sheet**
  - Write anything you want on it
  - Turn it in with your exam

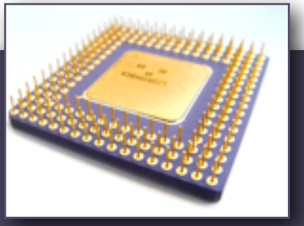
Administrivia: Final Exam



- **Friday, December 12, 12:00–2:30 p.m.**
- **Allowed one double-sided 8½×11” cheat sheet**

  - Write anything you want on it
  - Turn it in with your exam
- Comprehensive – Material from Exams 1 & 2 (70%):
  - Review study guides
  - Review the exams themselves – expect similar questions

Administrivia: Final Exam



- **Friday, December 12, 12:00–2:30 p.m.**
- **Allowed one double-sided 8½×11" cheat sheet**

  - Write anything you want on it
  - Turn it in with your exam
- Comprehensive – Material from Exams 1 & 2 (70%):
  - Review study guides
  - Review the exams themselves – expect similar questions
- New topics since Exam 2 (30%): floating-point, heap memory, caching/memory hierarchy
  - Review in-class activities – there *will* be questions like those on the activity sheets
  - Review the assigned reading, especially on cache memory
  - Study guide for this material will be posted

I'M ON A BOAT	MO MONEY MO PROBLEMS	IMOGEN'S SPARKS	CALL ME MAYBE
$100	$100	$100	$100
$200	$200	$200	$200
$300	$300	$300	$300
$400	$400	$400	$400
$500	$500	$500	$500

Final Jeopardy

1 - $100

THIS IS THE NUMBER OF BITS IN THE IEEE 754 DOUBLE-PRECISION REPRESENTATION OF A FLOATING-POINT NUMBER.

1 - $100

THIS IS THE NUMBER OF BITS IN THE IEEE 754 DOUBLE-PRECISION REPRESENTATION OF A FLOATING-POINT NUMBER.

What is 64?

1 - $200

THIS IS THE MASM DATA TYPE USED TO DEFINE SINGLE-PRECISION FLOATING-POINT NUMBERS.

1 - $200

THIS IS THE MASM DATA TYPE USED TO DEFINE SINGLE-PRECISION FLOATING-POINT NUMBERS.

What is REAL4?

1 - $300

THIS INSTRUCTION LOADS A FLOATING-POINT VALUE FROM MEMORY, PUSHING IT ONTO THE FLOATING-POINT STACK AT ST(0)

1 - $300

THIS INSTRUCTION LOADS A FLOATING-POINT VALUE FROM MEMORY, PUSHING IT ONTO THE FLOATING-POINT STACK AT ST(0)

What is FLD?

1 - $400

IF A .DATA SECTION CONTAINS

```
TOO   REAL4  2.0
TREE  REAL4  3.0
```

THIS IS THE VALUE IN ST(0) AFTER EXECUTING

```
FLD   TOO
FLD  TREE
FSUB
```

1 - $400

IF A .DATA SECTION CONTAINS

```
TOO   REAL4  2.0
TREE  REAL4  3.0
```

THIS IS THE VALUE IN ST(0) AFTER EXECUTING

```
FLD   TOO
FLD  TREE
FSUB
```

What is –1.0?

1 - $500

WHEN INTERPRETED AS A SINGLE-PRECISION FLOATING-POINT NUMBER, THE 32 BITS

B F C 0 0 0 0 0 h

1011 1111 1100 0000 0000 0000 0000 0000 b

REPRESENT THIS VALUE.

1 - $500

WHEN INTERPRETED AS A SINGLE-PRECISION FLOATING-POINT NUMBER, THE 32 BITS

B F C 0 0 0 0 0 h

1011 1111 1100 0000 0000 0000 0000 0000 b

REPRESENT THIS VALUE.

What is –1.5?

2 - $100

THIS PRINCIPLE STATES THAT INSTRUCTIONS EXECUTED WITHIN A SHORT PERIOD OF TIME TEND TO BE CLOSE TOGETHER IN MEMORY, AND DATA THAT ARE ACCESSED WITHIN A SHORT PERIOD OF TIME ALSO TEND TO BE CLOSE TOGETHER IN MEMORY.

2 - $100

THIS PRINCIPLE STATES THAT INSTRUCTIONS EXECUTED WITHIN A SHORT PERIOD OF TIME TEND TO BE CLOSE TOGETHER IN MEMORY, AND DATA THAT ARE ACCESSED WITHIN A SHORT PERIOD OF TIME ALSO TEND TO BE CLOSE TOGETHER IN MEMORY.

What is the Principle of Locality?

2 - $200

THIS IS AN ORGANIZATION OF STORAGE DEVICES THAT TAKES ADVANTAGE OF THE CHARACTERISTICS OF DIFFERENT STORAGE TECHNOLOGIES TO IMPROVE THE OVERALL PERFORMANCE OF A COMPUTER SYSTEM.

2 - $200

THIS IS AN ORGANIZATION OF STORAGE DEVICES THAT TAKES ADVANTAGE OF THE CHARACTERISTICS OF DIFFERENT STORAGE TECHNOLOGIES TO IMPROVE THE OVERALL PERFORMANCE OF A COMPUTER SYSTEM.

What is a memory hierarchy?

2 - $300

IF A 2-WAY SET ASSOCIATIVE CACHE HAS 8 ENTRIES, THIS IS THE NUMBER OF ENTRIES IN WHICH A PARTICULAR BLOCK OF MEMORY MAY BE STORED.

2 - $300

IF A 2-WAY SET ASSOCIATIVE CACHE HAS 8 ENTRIES, THIS IS THE NUMBER OF ENTRIES IN WHICH A PARTICULAR BLOCK OF MEMORY MAY BE STORED.

What is 2?

(Every 2 rows forms a set, and each block must be stored in one particular set)

2 - $400

IF MEMORY ADDRESSES ARE 32 BITS,
AND A CACHE HAS 64-BYTE CACHE LINES,
THIS MANY BITS OF A MEMORY ADDRESS
WILL BE USED TO IDENTIFY THE BLOCK NUMBER.

2 - $400

IF MEMORY ADDRESSES ARE 32 BITS, AND A CACHE HAS 64-BYTE CACHE LINES, THIS MANY BITS OF A MEMORY ADDRESS WILL BE USED TO IDENTIFY THE BLOCK NUMBER.

What is 26?

(64 = 2^6, so the low 6 bits identify the offset within a block,
and the upper 32 – 6 = 26 bits identify the block number)

2 - $500

IN A 2-WAY SET ASSOCIATIVE CACHE,
IF 26 BITS OF A MEMORY ADDRESS ARE USED TO IDENTIFY THE BLOCK NUMBER,
THIS MANY BITS ARE USED FOR THE TAG.

2 - $500

**IN A 2-WAY SET ASSOCIATIVE CACHE,
IF 26 BITS OF A MEMORY ADDRESS ARE USED TO
IDENTIFY THE BLOCK NUMBER,
THIS MANY BITS ARE USED FOR THE TAG.**

What is 25?

(The lowest 1 bit identifes the set in the cache; the remaining 26 – 1 = 25 are the tag)

3 - $100

THIS IS A MEMORY POOL FOR A SPECIFIC PROCESS. FROM WHICH MEMORY CAN BE ALLOCATED DYNAMICALLY. ITS SIZE IS NOT FIXED AND IS GENERALLY LARGER THAN THE STACK.

3 - $100

THIS IS A MEMORY POOL FOR A SPECIFIC PROCESS. FROM WHICH MEMORY CAN BE ALLOCATED DYNAMICALLY. ITS SIZE IS NOT FIXED AND IS GENERALLY LARGER THAN THE STACK.

What is the heap?

3 - $200

THIS WIN32 API FUNCTION IS USED TO ALLOCATE MEMORY ON THE HEAP.

3 - $200

THIS WIN32 API FUNCTION IS USED TO ALLOCATE MEMORY ON THE HEAP.

What is HeapAlloc?

3 - $300

IF HEAPALLOC IS UNABLE TO ALLOCATE MEMORY, IT RETURNS THIS VALUE

3 - $300

IF HeapAlloc IS UNABLE TO ALLOCATE MEMORY, IT RETURNS THIS VALUE

What is 0?

3 - $400

X86 PROCESSSORS BOOT INTHIS MODE, WHICH USES 20-BIT MEMORY ADDRESSES

3 - $400

X86 PROCESSSORS BOOT INTHIS MODE, WHICH USES 20-BIT MEMORY ADDRESSES

What is real-address mode?

3 - $500

ALTHOUGH YOUR PROGRAM'S DATA BEGINS AT MEMORY ADDRESS 00405000H, THAT IS NOT A PHYSICAL MEMORY ADDRESS; IT IS THIS TYPE OF MEMORY ADDRESS.

3 - $500

ALTHOUGH YOUR PROGRAM'S DATA BEGINS AT MEMORY ADDRESS 00405000H, THAT IS NOT A PHYSICAL MEMORY ADDRESS; IT IS THIS TYPE OF MEMORY ADDRESS.

What is a virtual address?

4 - $100

THIS INSTRUCTION POPS A DWORD OFF OF THE STACK, THEN JUMPS TO THE INSTRUCTION AT THAT MEMORY ADDRESS

4 - $100

THIS INSTRUCTION POPS A DWORD OFF OF THE STACK, THEN JUMPS TO THE INSTRUCTION AT THAT MEMORY ADDRESS

What is RET?

4 - $200

IN THIS CALLING CONVENTION,
THE CALLER
IS RESPONSIBLE FOR REMOVING
ARGUMENTS FROM THE STACK

4 - $200

**IN THIS CALLING CONVENTION,
THE CALLER
IS RESPONSIBLE FOR REMOVING
ARGUMENTS FROM THE STACK**

What is the C calling convention?

4 - $300

AFTER A STACK FRAME HAS BEEN CREATED, THE FIRST ARGUMENT TO A FUNCTION WILL BE FOUND AT THE MEMORY ADDRESS EBP + THIS VALUE

4 - $300

AFTER A STACK FRAME HAS BEEN CREATED, THE FIRST ARGUMENT TO A FUNCTION WILL BE FOUND AT THE MEMORY ADDRESS EBP + THIS VALUE

What is 8?

4 - $400

IF THE VERY FIRST INSTRUCTION IN A PROCEDURE IS

```
MOV EAX, [EBP+8],
```

YOU PROBABLY MADE A MISTAKE. BEFOREHAND, YOU SHOULD HAVE INSERTED THIS INSTRUCTION.

4 - $400

IF THE VERY FIRST INSTRUCTION IN A PROCEDURE IS

```
MOV EAX, [EBP+8],
```

YOU PROBABLY MADE A MISTAKE. BEFOREHAND, YOU SHOULD HAVE INSERTED THIS INSTRUCTION.

What is ENTER?

4 - $500

THE INSTRUCTION

`RET 8`

INCREASES ESP BY THIS AMOUNT, *IN TOTAL*

4 - $500

THE INSTRUCTION
`RET 8`
INCREASES ESP BY THIS AMOUNT, *IN TOTAL*

What is 12?

(First, it pops a 4-byte return address. Then, it removes 8 bytes of arguments.)