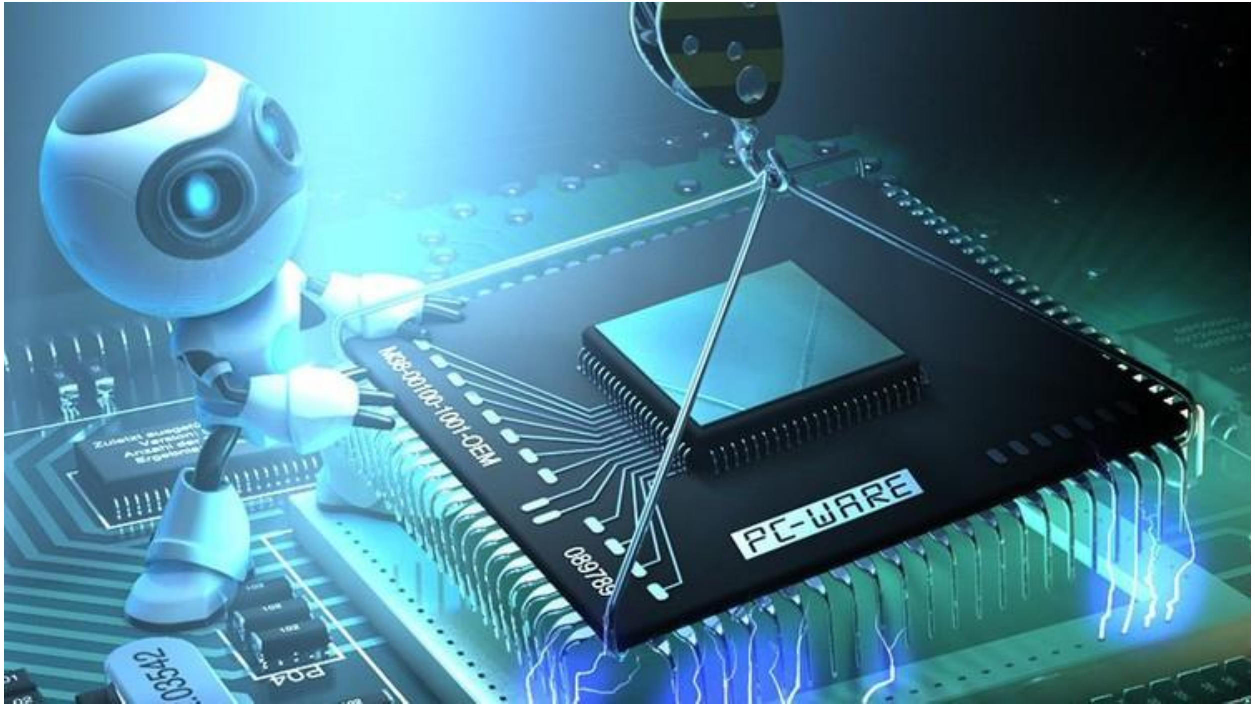
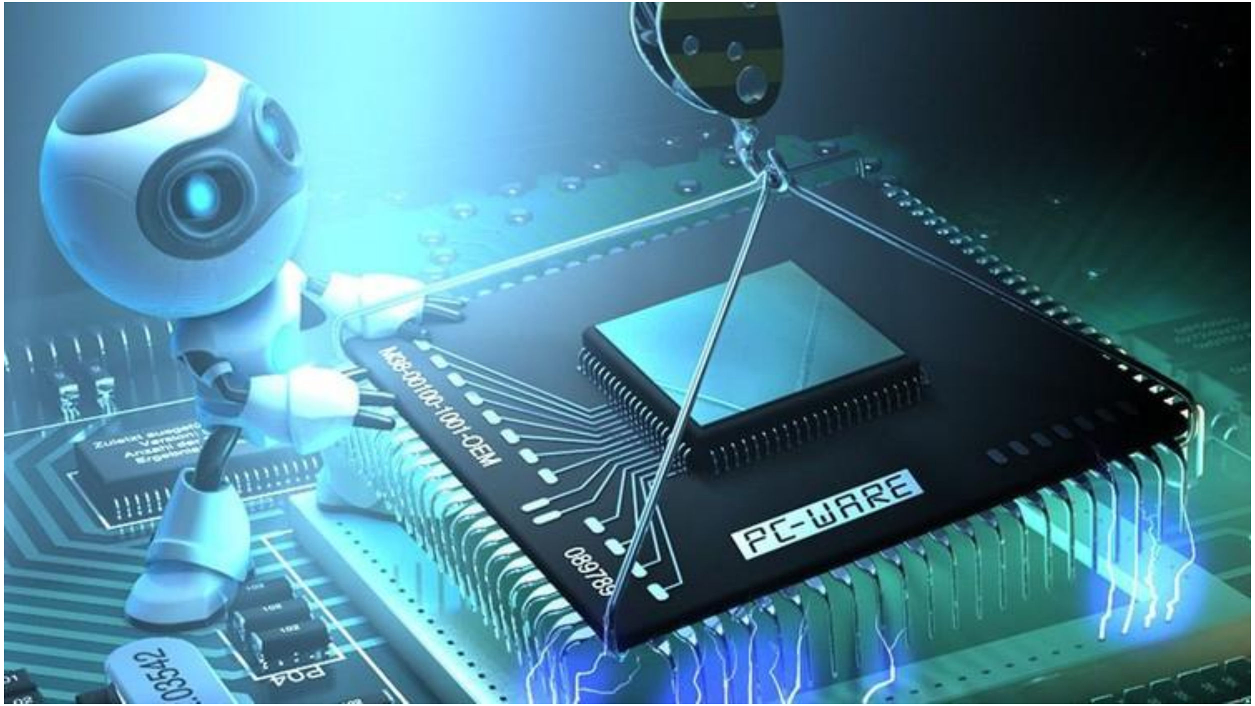
Exam 1

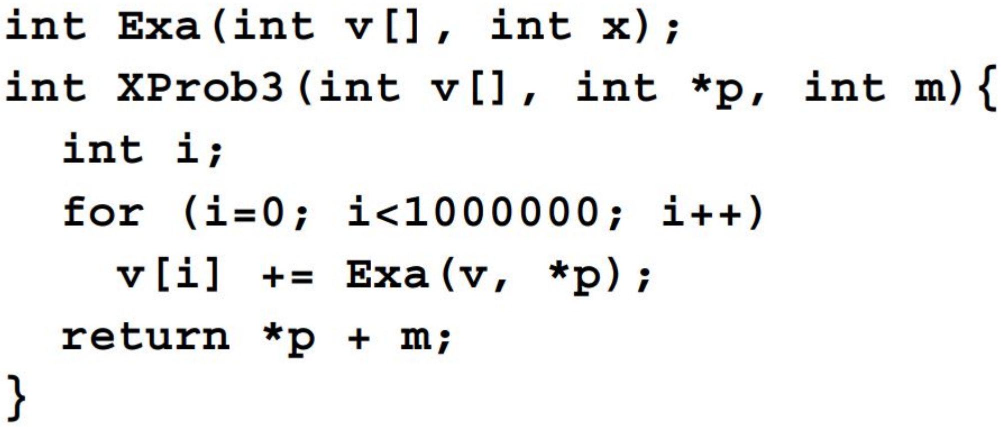


Exam 1 - Problem 2



Exam 1 - Problem 2

Given the following code written in C:



```
int Exa(int v[], int x);
int XProb3(int v[], int *p, int m){
  int i;
  for (i=0; i<1000000; i++)
    v[i] += Exa(v, *p);
  return *p + m;
}
```

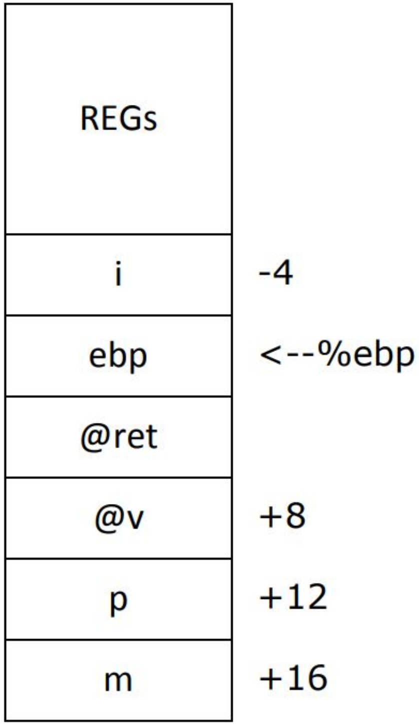
Exam 1 - Problem 2

a) **Draw** the activation block of the Xprob3 subroutine.

b) **Translate** the Xprob3 subroutine to x86 assembler.

Exam 1 - Problem 2

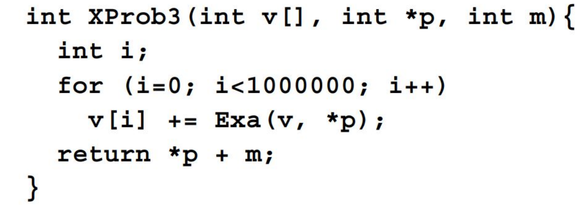
a) **Draw** the activation block of the Xprob3 subroutine.



REGs	
i	-4
ebp	<--%ebp
@ret	
@v	+8
p	+12
m	+16

Exam 1 - Problem 2

b) **Translate** the Xprob3 subroutine to x86 assembler.



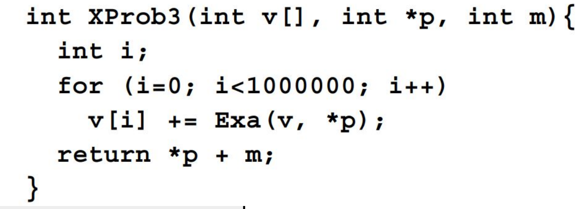
```
int XProb3(int v[], int *p, int m){
  int i;
  for (i=0; i<1000000; i++)
    v[i] += Exa(v, *p);
  return *p + m;
}
```

```
Xprob3:
 pushl %ebp
 movl %esp,%ebp
 subl $4, %esp              # only one local variable i
 pushl %esi                 # esi will represent i
 pushl %ebx
 movl 8(%ebp),%ebx          # ebx = @v
 xorl %esi,%esi             # i = 0
for: cmpl $1000000, %esi
 jge endfor                 # we jump to endfor if i >= 1000000
 movl 12(%ebp), %eax        # eax = &p
 pushl (%eax)               # second argument is *p
 pushl %ebx                 # first argument is v
 call Exa
```

Part 1/2

Exam 1 - Problem 2

b) **Translate** the Xprob3 subroutine to x86 assembler.



```
int XProb3(int v[], int *p, int m){
  int i;
  for (i=0; i<1000000; i++)
    v[i] += Exa(v, *p);
  return *p + m;
}
```

```
 addl $8, %esp
 addl %eax, (%ebx, %esi, 4)        # v[i] = v[i] + result
 incl %esi                         # i++
 jmp for
endfor:
 movl 12(%ebp), %eax               # eax = p
 movl (%eax), %eax                 # eax = *p
 addl 16(%ebp), %eax               # eax = *p + m
 popl %ebx
 popl %esi
 movl %ebp, %esp
 popl %ebp
 ret
```

Part 2/2