1. a) The time distribution t_i is different from the frequency distribution; it is the ratio between the time T_i the instruction i consumes in the workload, and the total time, CPU_{time}:

$$t_i = \frac{T_i}{CPU_{time}}$$

Start with the relation for the CPU_{time}:

$$CPU_{time} = CPI \cdot T_{ck} \cdot IC$$

$$CPU_{time} = CPI \cdot T_{ck} \cdot \sum_i IC_i$$

where IC_i is the instruction count for instruction i. This can be expressed as a function of T_i, CPI_i and T_{ck}:

$$T_i = IC_i \cdot CPI_i \cdot T_{ck}$$

It yields from the last two relations:

$$CPU_{time} = CPI \cdot T_{ck} \cdot \sum_i \frac{T_i}{CPI_i \cdot T_{ck}}$$

$$\frac{1}{CPI} = \sum_i \frac{T_i}{CPU_{time} \cdot CPI_i}$$

from which we get the final relation:

$$\frac{1}{CPI} = \sum_i \frac{t_i}{CPI_i}$$

Using the given values yields:

$$\frac{1}{CPI} = \frac{0.5}{4} + \frac{0.4}{6} + \frac{0.1}{5} = 0.211 \Rightarrow CPI = 4.73$$

b) The relation to compute MIPS is:

$$MIPS = \frac{IC}{CPU_{time} \cdot 10^6} = \frac{IC}{IC \cdot CPI \cdot T_{ck} \cdot 10^6} = \frac{f_{ck}}{CPI \cdot 10^6}$$

$$MIPS = \frac{200 \cdot 10^6}{4.73 \cdot 10^6} = 42.2$$

2. a) You need to convert the decimal address to its hexadecimal representation; to do so use the division algorithm. It results:

$(268500976)_{10}$ = 0x1000fff0

The sequence of code follows:

```
lui $1, 0x1000          # the upper part of the array address
ori $1, $1, 0xfff0      # the full address is now in $1
lw $t0, 24($1)          # load x[6]
add $t0, $t0, $t1       # x[6] + a
sw $t0, 20($1)          # store at x[5] whose displacement is 20.
```

b) The sequence of addresses issued by the CPU is presented below:

Address	R/W	Comment
0x400018	R	fetch: `lui $1, 0x1000`
0x40001c	R	fetch: `ori $1, $1, 0xfff0`
0x400020	R	fetch: `lw $t0, 24($1)`
0x10010014	R	load: `x[6]`
0x400024	R	fetch: `add $t0, $t0, $t1`
0x400028	R	fetch: `sw $t0, 20($1)`
0x10010010	W	store at: `x[5]`

3. a) Note that you may not apply the Amdahl's formula here: the reason is that we know the fraction of the running time (20%) with the improvment in use. The Amdahl's relation requires the fraction before the enhancement is made.

$$\frac{CPU_{time-old}}{CPU_{time-new}} = \frac{(1-f_{new}) \cdot CPU_{time-new} + f_{new} \cdot CPU_{time-new} \cdot S_0}{CPU_{time-new}}$$

$$S_{overall_a} = \frac{CPU_{time-old}}{CPU_{time-new}} = (1-f_{new}) + f_{new} \cdot S_0$$

$$S_{overall_a} = (1-0.2) + 0.2 \cdot 2 = 1.2$$

b) Assuming that two independent enhancements are made at the same time, and that f_1 and f_2 fractions of the initial workload can benefit of it, we have

$$S_{overall_b} = \frac{1}{(1-(f_1+f_2)) + \frac{f_1}{S_{01}} + \frac{f_2}{S_{02}}}$$

with S_{01} and S_{02} the speedups for each individual enhancement.
The fraction f_1 can be calculated using the result in part a:

$$S_{overall_a} = \frac{1}{(1-f_1) + \frac{f_1}{S_{01}}}$$

$$f_1 = \frac{(S_{overall_a} - 1)}{\left(S_{overall_a} - \frac{S_{overall_a}}{S_{01}}\right)}$$

$$f_1 = \frac{(1.2 - 1)}{\left(1.2 - \frac{1.2}{2}\right)} = \frac{1}{3} = 33.3\%$$

Then the overall speedup with both enhancements in place is

$$S_{overall_b} = \frac{1}{(1 - 0.333 - 0.30) + \frac{0.333}{2} + \frac{0.30}{20}} = 1.82$$

4. a) assume that a pointer has the same size as a word. Then an instruction like

```
lw R1, @(R2)
```

will be replaced by a sequence of two instructions:

```
lw R1, 0(R2)
lw R1, 0(R2)
```

b)

$$CPU_{time-new} = IC_{new} \cdot CPI \cdot T_{ck-new}$$

$$IC_{new} = IC_{old} + f_{indirect} \cdot 1 \cdot IC_{old}$$

because for every memory indirect reference in the old program there will be one more in the new program.

$$IC_{new} = IC_{old} \cdot (1 + 0.05) = 1.05 \cdot IC_{old}$$

$$T_{ck-new} = 0.9 \cdot T_{ck-old}$$

$$\frac{CPU_{time-old}}{CPU_{time-new}} = \frac{IC_{old} \cdot CPI \cdot T_{ck-old}}{IC_{new} \cdot CPI \cdot T_{ck-new}}$$

$$\frac{CPU_{time-old}}{CPU_{time-new}} = \frac{IC_{old} \cdot CPI \cdot T_{ck-old}}{1.05 \cdot IC_{old} \cdot CPI \cdot 0.9 \cdot T_{ck-old}} = \frac{1}{1.05 \cdot 0.9} = 1.058$$

The new architecture is by 5.8%faster than the old one.