1　アルゴリズムとプログラミング

(1) 2 巡目に入るとき、buckets の値は初期化されないことに注意

buckets[x][y]	1 巡目 (d=1)					2 巡目 (d=2)				
	y=0	y=1	y=2	y=3	y=4	y=0	y=1	y=2	y=3	y=4
x=0	0	0	0	0	0	1	2	0	0	0
x=1	21	1	11	0	0	11	12	11	0	0
x=2	12	2	0	0	0	21	2	0	0	0

(2-1) 基数ソート

(2-2) $O(kn)$

n 個のデータに対しての処理を各桁で、すなわち k 回行う。

(2-3) 各桁を最下位ビットから見ていく。各ビットの値を小さいを順に格納していく。値が同じである場合は安定なソートアルゴリズムであるため下位ビットの大小関係を保持できるのでソートが可能である。

(2-4) 整列対象のデータ同士の比較を伴う整列アルゴリズムでは下界のアルゴリズムでも $O(n \log n)$ であり、それよりデータ数 n について時間計算量が少ない。しかし、$r * n$ の大きさの配列が必要となる。

(3) (ア) $b = r - 1; b >= 0; b - -$

(イ) $j = 0; j < numbucket[b]; j + +;$

2　計算機システムとシステムプログラム

(1-1) (a) エ (b) コ (c) ウ (d) イ (e) カ

(1-2-1) $\frac{mn}{f}$

(1-2-2) $\frac{m+n-1}{f}$

(1-2-3) f

(1-2-4) ステージ数を増やすことで 1 クロックサイクルあたりに同時に実行される命令数が増えるため

(1-3-1)	クロックサイクル	0	1	2	3	4	5	6	7	8	9	10	11
	命令 1 : MOV R1,(A)	IF	D	OF	EX	S							
	命令 2 : MOV R2,(B)		IF	D	OF	EX	S						
	命令 3 : ADD R1,R2			→	→	IF	D	OF	EX	S			
(1-3-2)	クロックサイクル	0	1	2	3	4	5	6	7	8	9	10	11
	命令 1 : MOV R1,(A)	IF	D	OF	EX	S							
	命令 2 : INC R1		IF	D	→	→	OF	EX	S				
	命令 2 : MOV (B),R1			→	IF	D	→	→	→	OF	EX	S	

(1-3)

(2-1-1) $2 \times \frac{80}{100} + 50 \times \frac{20}{100} = 11.6$

(2-1-2) $2 \times \frac{x}{100} + 50 \times \frac{100-x}{100} = 11.6$

$x = 83.5\%$

(2-2-1) アクセスされたデータの近傍のデータが近い将来アクセスされる可能性が高いこと

(2-2-2) アクセスされたデータが近い将来再度アクセスされる可能性が高いこと

(2-3) 参照局所性の観点からアクセスされる可能性が高いデータを主記憶よりもアクセス速度が速いキャッシュメモリに配置することで、平均アクセス時間が短くなる点