〔韓國造景學會誌, Vol. 18, No. 4 pp. 17~28 1991年 1月〕

都市 오픈스페이스의 接近性 測定에 관한 硏究

安東晩* · 崔亨碩** · 金仁浩*** · 趙亨濬****

* 서울대학교 조경학과
** 수원대학교 도시공학과
*** 동인조경
**** 서울대학교 대학원 생태조경학과

A Study on the Method of Measuring Accessibility to Urban Open Spaces

Ahn, Tong－Mahn*. Choi, Hyung－Seok. Kim, In－Ho***. Cho, Hyoung－Jun******

* Dept. of Landscape Architecture, College of Agri. Seoul National Univ.
** Dept. of Urban Engineering, College of Engineering, Suwon Univ.
*** Tong－In Landscape Architecture Co.
**** Master of Landscape Architecture, College of Agri. Seoul National Univ.

ABSTRACT

The purpose of this study is to investigate and present a method for measuring public accessibility to urban open spaces.

A basic assumption is that, for urban open space policies, accessibility is more important than per capita area. In this study, for the purpose of simplicity, a residential area is assumed to have access to open space if it is within a certain distance from an urban open space. Official city planning map is overlayed with a 200m grid and each cell of dwelling area is checked whether it is within a certain distance from a cell categorized as urban open space. A computer program for widely commercialized personal computer is developed for data processing so that local governments without access to more sophisticated systems can carry out similar studies for their own jurisdictions.

Five cities, big, small, old and new, are selected to test the proposed method. Dwelling areas of Ansan new Town have highest accessibility to open spaces(93.4% of dwelling cells have open space cell within 500m). Seoul (91.2%), Suwon(78.2%), Pusan(73.8%), and Inchon(61.4%)have less accessibility.

If we assume the Ansan City residents are evenly distributed over the dwelling areas, 93.4% of the population has open spaces within walking distance of 500m. However, if we consider physical barriers such as arterial roads, railroads, and streams that reduce the accessibility, less than 93.4% of Ansan city residents enjoy good access to open spaces.

Though a further detailed analysis is needed to picture the microscopic accessibility, this method can serve as a useful tool for urban open space policy and open space alternatives evaluations.

Ⅰ. 서 론

1. 연구의 배경 및 목적

근대 이후 가속화된 都市의 産業化는 더욱 도시로의 인구집중을 유발하여 人口過密의 문제와 더불어 都市환경의 惡化라는 결과를 招來하게 되었다.

1960년대 이후 꾸준한 경제성장을 지속해온 우리나라는 都市化 또한 더욱 급속하게 진행되어 인구 및 산업의 팽창이 都市基盤施設의 수용능력을 초과하게 됨에 따라 都市開發에 있어서 균형적 발전을 저해하게 되었고 그 과정에서 시민생활의 質에 대한 고려는 微弱했다고 볼 수 있다.

이러한 都市內에서 休息과 娛樂의 場을 공급하고 人工 및 自然災害에 대한 피난처를 제공하며 都市美化의 역할을 담당하는 오픈 스페이스(OPEN SPACE)의 기능은 現代도시에 있어서 매우 중요하게 인식되고 있으며, 더우기 所得水準의 향상 및 이에 따른 餘暇時間의 증가는 屋外活動에 대한 需要를 증대시키고 있는 바(서울특별시, 1985), 시민들에 의하여 潛在的으로 요구되어지는 수요를 충족시키기 위한 장소로서 오픈스페이스는 그 確保의 중요성이 切實하다고 하겠다.

오픈스페이스는 개념을 정의할 때 일반적으로 개발지나 비건폐지, 위요공간, 그리고 자연환경 등의 형태적 측면으로 구분되나(황 외, 1989)현대에 있어서는 이러한 형태적 측면 외에도 기능적, 행태적 측면에서의 고찰도 중요하다고 인식되는 바, Kevin Lynch가 언급한 '오픈스페이스의 이용자에 대한 개방성(OPENNESS)'은 (Lynch,1971)오픈스페이스가 면적, 공간의 특성보다는 이용적 측면에서의 접근성이 중요함을 강조한 것이라 볼 수 있다.

그러나 도시의 土地利用上 지금까지 진행되어온 개발의 樣態를 살펴보면 오픈 스페이스는 關聯法規에서 요구되어지는 量的 確保 및 供給에만 그치고있어 需要에 대한 양적 측면과 더불어 효용성 측면에 대한 고려 또한 아쉬운 실정이다.

실제로 서울의 경우 전국토의 0.63%밖에 안되는 면적에 전 인구의 23.4%에 달하는 970만명이 살고 있으나 平均公園面積이 市民 1인당 5.4㎡로 東洋圈에서 볼 때 양적 측면에서의 공급수준은 결코 낮은 편이 아니라 할 수 있다. 그러나 전체 공원중 시민이 쉽게 接近할 수 있는 공원의 면적은 狹小하고, 접근이 어려운 자연공원이 70%를 차지하고 있으며 또한 近隣公園도 대부분 도로에서 떨어진 野山에 위치하고 있어서 실제로는 시민의 효율적 이용이나 도시내에서의 접근이 容易하지 않다. (서울특별시, 1985) 利用便益의 측면에서 볼 때 기존의 OPEN SPACE조차 충분히 개발, 활용되지 못하고 있고 대체로 거리가 멀어서 生活空間과 밀접하게 연관되어 있지 않다는 것은 양적인 부족못지않게 중요한 문제임에 틀림없다.(이, 1985)

이러한 결과는 公園綠地體系 수립 時 기존의 接近方法이 단순히 일인당 공원면적 및 공원률만을 계산하여 고려하며, 오픈스페이스까지의 접근거리를 계획에 반영할 수 있는 체계적인 접근이 부족하였기 때문이라 볼 수 있다.

이에 本稿에서는 近隣住區 生活圈내에서의 오픈스페이스는 도시민이 쉽게 접근할 수 있고 주민생활에 밀접하게 연관되어야 한다는 접근성(ACCESSIBILITY)의 개념에 의거하여 거시적인 측면에서 도시간 또는 도시내 지역간 오픈스페이스의 접근성을 측정하는 방법을 제시하는 한편 실제로 측정과정을 통하여 적용을 시도함으로써 정책입안의 자료로서 오픈스페이스 공급 우선순위판정의 근거를 마련할 수 있도록 한다.

2. 연구의 방법

文獻硏究를 통하여, 첫째 오픈 스페이스의 槪念, 分類體系 및 기능을 살펴봄으로써 오픈 스페이스에 대한 접근성 측정의 필요성을 인식하는 한편 접근성 측정방법의 개발에 있어서 도시계획도상의 용도중 오픈 스페이스로 구분할 기준을 채택하고, 둘째 기존의 도시인문지리학 및 도시교통학에서 연구되어온 접근성에 관한 내용을 정리하여 접근성에 관한 개념 및 측정방법의 기본적 틀을 설정하였다.

접근성 측정의 방법에 있어서, 첫째 대상지에 대한 자료입력에는 1/25,000 都市計劃圖를 사용하여 200m × 200m 격자(GRID)로 구획한 다음 4가지 용도로 크게 구분하였고, 둘째 접근성 측정을 위한 測定變數의 선정은 직선거리로 하였다.

접근성 측정의 대상지로는 서울, 부산, 인천, 수원 그리고 안산 등 대도시 및 중소도시를 선정하고 이들 5개 도시의 오픈스페이스에 대한 면적 비율 및 접근성수준을 측정하여 측정결과를 상호 比較評價하였다.

이때 접근성 측정 및 측정 결과의 자료분석에는 IBM Personal Computer에서 C Language로 작성한 Program을 이용하여 작업하였다.(부록참조)

Ⅱ. 본 론

1. 접근성 측정방법

(1) 접근성 개념의 고찰

接近性은 人文地理學에서 공간적 현상의 多樣性 표현에 자주 사용되는 어휘이며 都市地理學에서는 도시의 成長을 설명할 때 便益施設의 위치 및 기능성의 문제와 土地利用配置에 관한 문제에 자주 사용되는 의미로서, 두 지점간의 近接의 측정을 포함하여 도달되어지는 정도를 나타낸다.

Ingram(1971)은 접근성을 相對的인 접근성(Relative Accessibility)과 統合的인 접근성(Integral Accessibility)으로 구분하여 설명하였다. 여기서 상대적 접근성은 같은 지면으로 연결된 두 지점의 상호 연결된 정도'라고 정의되어질 수 있으며, 통합적 접근성은 '같은 지면상에 위치한 모든 다른 지점과 주어진 한 지점이 상호 연결된 정도'로 정의된다.

都市交通體系에서 접근성은 일반적으로 무엇을 찾는데 필요한 노력의 쉬운 정도(Ease of Reaching)로 정의되며 유사한 의미로서의 移動性이 어느 한 위치에 있는 한 개인 또는 집단이 다른 곳으로 움직일 수 있는 능력을 의미하는데 반하여 접근성은 어느 활동을 수행할 수 있는 機會(Opportunity) 또는 潛在性(Potential)을 나타내고, 특히 物的體系의 양대 측면인 교통과 토지이용을 하나의 체계로 보고 이들에 대한 統合的 접근을 지향한다는 점에서 意義가 크다.(임, 1986)

그러나 접근성은 經濟性, 즉 費用의 문제와 또는 다른 지역간의 거리를 극복하게 하는 교통체계의 능력에 좌우된다고 할 수 있기 때문에 計量的 기초 위에서 그 자체의 복합적인 측정이 수행되어야 하며 따라서 개념의 특성을 정의하는 데는 어려움이 많다.(Baxter, 1974)

그러므로 접근성에 관한 기존의 개념 설정들이 연구의 기본적 틀은 제공하지만 시각을 달리한 새로운 개념의 정립이 필요하다고 인식된다. 따라서 본 연구에서는 접근성의 개념을 단순히 물리적인 측면에서 볼 때 '同一 平面上의 한 지점으로 부터 一定距離안에 접근을 목적으로 하는 지점이 存在함'이 라고 정의하고 연구를 진행하였다.

(2) 접근성 측정방법

(가) 자료의 입력

1/25,000의 도시계획도를 기준으로 하여 각 도시를 200m×200m(축척 1/25000에서 8㎜×8㎜)의 격자로 나눈다.

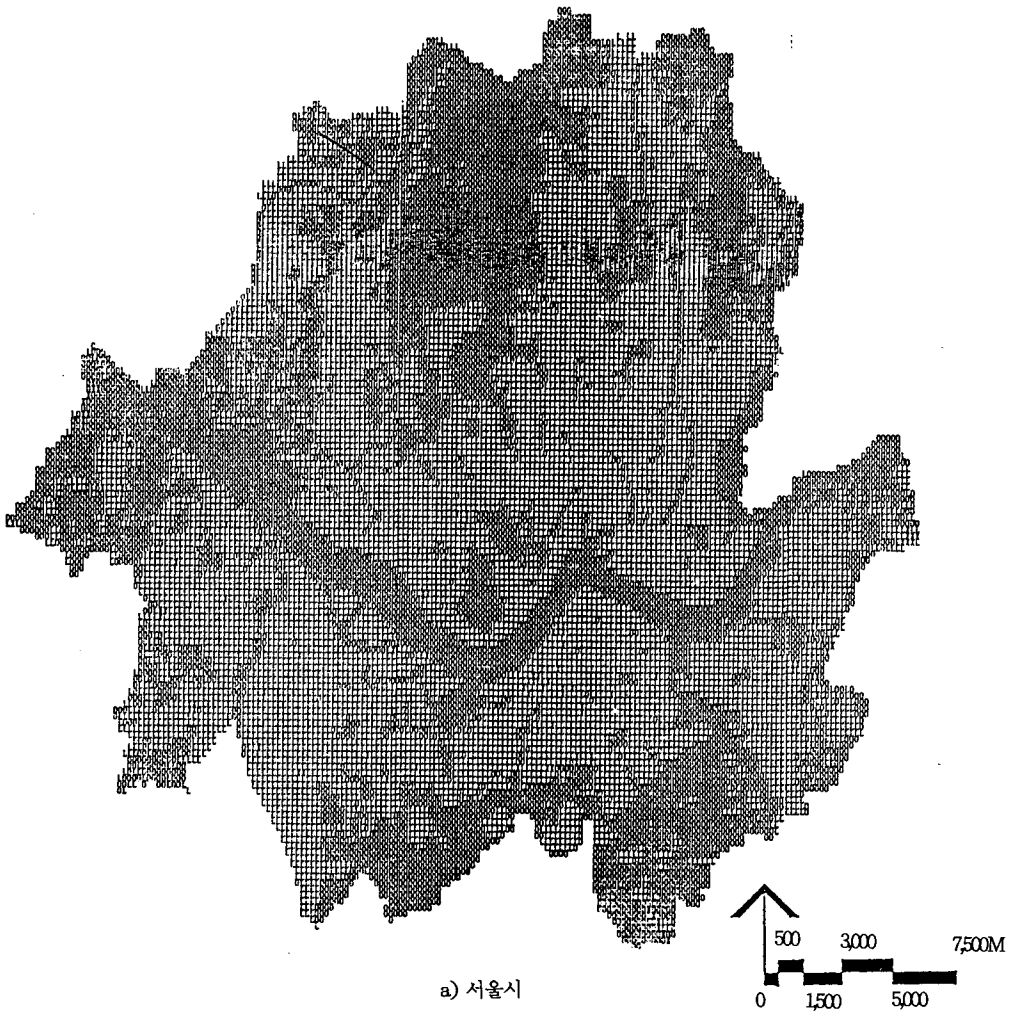
각 격자는 住居地 1번, 오픈스페이스 2번, 非住居地3번 그리고 계획대상지 밖은 0번 등으로 분류하였고 도시계획도 상의 凡例를 이상의 4지역으로 구분하기 위하여 다음과 같은 기준으로 설정하였다.

도시계획도 상의 범례	분 류 내 용
주거지역 준주거지역 상업지역 노선상업 공업지역 준공업지역 풍치지구	1번(주거)
자연녹지지역 생산녹지지역 시설녹지지역 개발제한구역 공원용지 유원지 운동장 하천 및 호수	2번(오픈스페이스)
철도 도로광장 50m 도로 고속국도	3번(비주거)
그외 계획도면 밖	0번

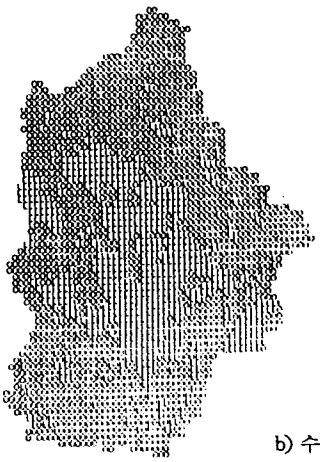
단, 生産綠地地域과 開發制限區域은 성격상 오픈스페이스로 분류하되 集成村이나 주거지로 개발되어지는 지역은 주거지로 분류했으며 한강고수부지와 여의도 광장, 그 외 호수의 慰樂과 餘暇를 즐길 수 있다고 판단되는 지역은 오픈스페이스로 분류하였다.

또한 1개의 격자에 대하여 여러 성격이 나타나는 경우에는 면적의 50% 이상을 차지하는 용도로 분류하였다.

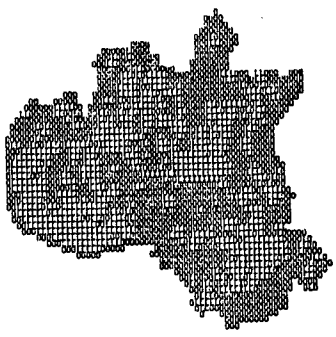
주거지, 오픈스페이스 그리고 비주거지는 資料入力시 각각 1, 2, 3번으로 분류되었으나 Computer출력상에서 資料出力시에는 "L","O","X"등으로 표시되도록 하였다(그림 1참조).



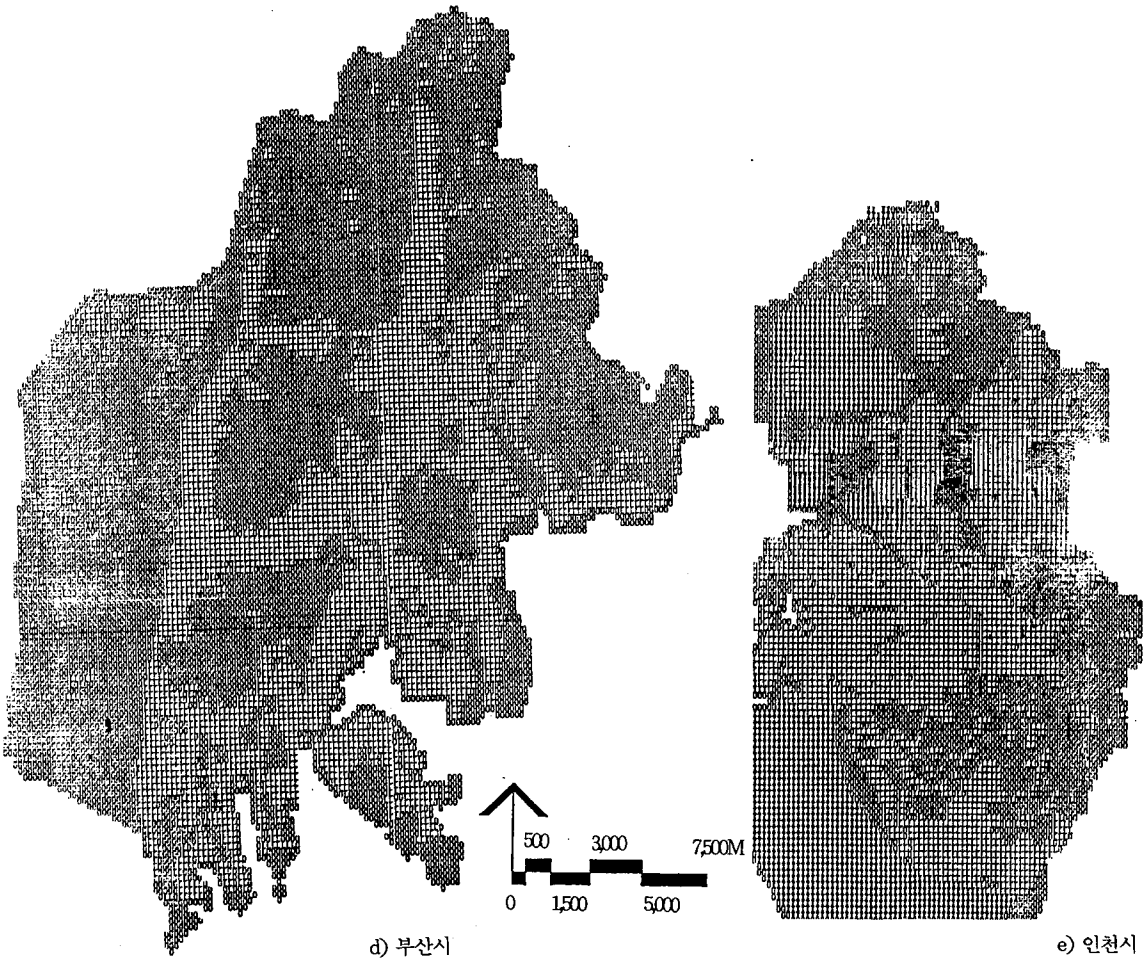
a) 서울시



b) 수원시



c) 안산시



(그림 1) 3분류에 의한 토지이용 구분

(나) 단계별 접근성에 따른 자료의 검색

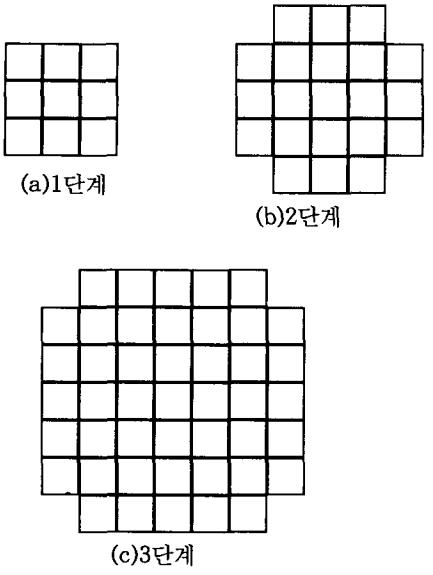
物理的 거리에 의하여 3단계로 접근성을 고려하였으며 여기서의 물리적 거리는 도로에 의한 거리가 아닌 주거지로 부터의 直線距離이다.

여기서의 직선거리는 近隣公園에 대한 유치거리를 근거로 하였다. 즉, 국내에서 근린공원을 4종으로 구분한 것중 유치거리 500m와 1,000m의 평균거리로 계산한 750m 및 미국의 기준인 1/2마일(윤, 1985), 그리고 보행시간과 보행거리를 고려하여 10−15분 정도의 거리를 통해 계산한 자료 그리고 이용자 측면에서 便益施設에 대한 접근성을 고려한 수치(양, 1985)등을 토대로 검토한 결과 800m를 한계로 설정하였으며 이를 기준으로 1단계, 2단계 그리고 3단계의 접근성 측정범위를 규정하였다.

각 단계에서 접근성 측정은 주거지로 분류된 격자를 중심으로 일정 거리내에 위치한 격자를 檢索하여 2번(오픈스페이스)격자가 있는 지의 여부를 판별하는 방법을 사용하였다. 이때 거리에 따른 검색 대상 격자의 한계를 정하는데는 다음과 같은 방법을 사용하여 격자의 위치에 따른 거리오차를 최소화하였다.

즉, 중심(주거지)격자의 좌표가 (X1, X2)라고 할 때, $\sqrt{(X2-X1)^2+(Y2-Y1)^2}\times 200$을 계산하여 계산결과가 일정거리(D : 300m, 500m,700m)를 넘지 않는 격자만을 검색하는 것이다.

이러한 방법을 적용하는 경우 1단계의 접근성 측정은 $\sqrt{(X2-X1)^2+(Y2-Y1)^2}$의 값이 1.5이하인 격자를 검색하게 되면 (그림2 a)의 경우 중심 셀 주위의 8개 셀이 검색될 것이다.



〔그림 2〕 단계별 검색 限界 격차

또한 2단계의 접근성 측정은 $\sqrt{(X2-X1)^2+(Y2-Y1)^2}$의 값이 2. 5 이하인 격자를 검색대상으로 하게 되며 〔그림 2b〕와 같이 중심 격자 주변의 20개 격자가 그 검색대상이 된다.

같은 방법으로 3단계의 접근성 측정시에는 $\sqrt{(X2-X1)^2+(Y2-Y1)^2}$의 값이 3.5이하인 격자를 검색대상으로 설정하여 검색하게 되며 중심격자 주변의 44개의 격자〔그림2c〕가 검색대상이 된다.

따라서 1단계의 접근성이란 중앙에 위치한 1번(주거지)격자의 중심으로 부터 300m내의 거리에 중심이 위치하는 모든 격자가 검색의 대상이 되며 이 격자들중에 2번(오픈스페이스)격자가 하나 이상 존재하느냐의 與否를 판단하는 것으로 이 경우는 1번(주거지)으로 분류된 격자주위에 있는 8개의 격자중에 2번(오픈스페이스)격자가 존재한다면 그 1번(주거지)격자는 오픈스페이스에의 접근성이 있다고 판단되며 3단계중 가장 높은 수준의 접근성을 의미한다.

마찬가지로 2단계와 3단계의 접근성이란 각각 1번(주거지)격자의 중심으로 부터 약 500m, 700m 반경내에 중심이 위치하는 격자중 2번(오픈스페이스)격자가 하나 이상 존재하는 지의 與否를 검색하는 것으로 1번(주거지)격자주위에 있는 20개(2단계), 44개(3단계)의 격자중에 2번(오픈스페이스)격자가 존재하면 중심의 1번(주거지)격자는 오픈스페이스에의 접근성이 있는 것으로 간주하였다.

2. 대상지를 통한 접근성 측정

서울, 부산, 인천, 수원 그리고 안산 등 5개 도시를 선정하여 오픈스페이스에 대한 접근성 측정 방법을 실제로 적용하고 측정결과를 상호 比較檢討하였다.

(1) 대상지선정

서울과 부산, 인천은 각각 特別市와 直轄市로 행정도시로서의 중요성 및 많은 用途의 지역이 혼재되어 있는 大都市의 성격을 대변하여 줄 수 있는 곳으로서 선택되었으며, 수원은 中小都市를 대표하여 대도시와의 비교를 위하여 선택되었다. 또한 안산은 최근에 새롭게 計劃建設되어지는 도시로서 이미 體系가 갖추어진 기존의 도시들과의 비교를 위하여 선정되었다.

(2) 대상지 접근성 측정 및 결과

1986년도에 刊行된 도시계획도를 토대로 자료를 입력하여 5개 도시들의 용도별 면적 비율 및 1번(주거지)격자의 오픈스페이스에 대한 접근성을 측정하였다. 입력된 토지분류 결과는 〈표1〉과 같다.

〈表 1〉 각 도시의 토지이용 분류

대상지	총격자수	주거지격자 (1번 분류)수	오픈스페이스격자 (2번 분류)수	비주거지격자 (3번 분류)수
서 울	17,007	9,131 (53.7)	7,732 (45.5)	144 (0.8)
부 산	11,978	3,989 (33.3)	7,953 (66.4)	36 (0.3)
인 천	7,204	2,823 (39.2)	2,885 (40.0)	1,496 (20.8)
수 원	2,434	821 (33.8)	1,610 (66.1)	3 (0.1)
안 산	1,973	724 (36.7)	1,223 (62.0)	26 (1.3)

각 결과치 아래의 괄호안에 표기된 數値는 도시별 총 격자수에 대한 각 용도별 분류된 격자수를 百分率로 나타낸 것이다.

여기서 총 격자수는 전체 분류된 격자중에서 0번(계획도면 밖)으로 분류된 격자의 數를 제외한 數値이며, 따라서 총 격자수는 대략적인 도시의 크기를 나타낸다.

총 격자수에 대한 2번(오픈스페이스)격자수의 비율을 보면 부산, 수원, 안산이 각각 66.4%, 66.1% 그리고 62.0%로 전체 도시 면적의 60%이상을 오픈스페이스가 점유하고 있는 것으로 나타나 서울의 45.5%나 인천의 40.0%에 비하여 상대적으로 높은 수치

를 보여주고 있다.

이러한 자료를 이용하여 각 단계별 접근성을 측정한 결과는 〈표 2〉와 같다.

〈表 2〉 오픈스페이스에 대한 접근성 측정결과

대상지	주거지격자(1번분류)수	1단계접근성 격자수(300m)	2단계접근성 격자수(500m)	3단계접근성 격자수(700m)
서 울	9,131	6,035 (66.1)	8,323 (91.2)	9,005 (98.6)
부 산	3,989	1,915 (48.0)	2,943 (73.8)	3,592 (90.0)
인 천	2,823	1,065 (37.7)	1,734 (61.4)	2,322 (82.3)
수 원	821	442 (53.8)	642 (78.2)	763 (92.9)
안 산	724	521 (72.8)	676 (93.4)	722 (99.7)

각 결과치 아래의 괄호안에 표기된 數値는 都市別 1번(주거지)으로 분류된 격자의 數中 각 단계에서 접근성이 있다고 판단된 격자의 수를 百分率로 나타낸 것이다.

1번(주거지)격자수에 대한 각 단계별 접근성이 있다고 판단된 1번 격자수의 비율을 측정해 본 결과 1, 2, 3단계 모두 5개 도시의 順位가 동일하게 안산이 가장 높고 그 다음 서울, 수원, 부산, 인천 등의 순서로 나타났다.

안산은 각 단계별 접근성이 있다고 판단된 1번(주거지)격자의 수가 1단계521개(72.8%), 2단계 676개(93.4%), 3단계 722개(99.7%) 등으로 나타나 특히 3단계의 접근성 측정에 있어서는 2개의 격자만이 접근성이 없음을 보여준다. 반면 인천은 1단계가 1,065개(37.7%), 2단계가 1,734개(61.4%), 3단계가 2,322개(82.3%)를 기록하여 도시별로 심한 隔差를 보여 주고 있다.

한편 5개 도시의 도시내 오픈스페이스 면적 비율 순위와 각 단계별 접근성이 있다고 판단된 격자수의 비율 순위를 비교하여 보았을 때 인천을 제외하고는 일치하지 않는 점으로 미루어 보아 도시내 오픈스페이스 면적 비율이 주거지로 부터 오픈스페이스에 대한 접근의 容易함 정도에 크게 영향을 못미치는 것으로 판단된다. 예를 들어 부산시는 확보된 오픈스페이스 면적은 크나(66.4%로 1위)실제 이용상 접근성이 큰 오픈스페이스는 매우 적음(3단계의 접근성 90.0%로 4위)을 나타내었다.

(3) 대상지 측정 결과의 해석

1단계 오픈스페이스에의 접근성 측정 결과 직선거리 300m내에 오픈스페이스를 가지고 있는 주거 및 준주거지역, 상업지역, 노선상업, 공업 및 준공업지역, 풍치지구 등의 면적비율이 서울 66.1%, 부산 48.0%, 인천37.7%, 수원 53.8%, 안산 72.8%등으로 나타나, 인구밀도가 균등하다고 가정하면 서울의 경우 66.1%의 시민이 300m내에 오픈스페이스를 가지고 있어 도보로 5분이내에 도달, 이용 가능한 것이다.

2,3단계 오픈스페이스에의 접근성 측정 결과는 각각 직선거리 500m, 700m내에 오픈스페이스를 가지고 있는 주거 및 준주거지역, 상업지역, 노선상업, 공업 및 준공업지역, 풍치지구 등의 면적 비율이 서울 91.2%(2단계) 98.6%(3단계), 부산 73.8%(2단계) 90.0%(3단계), 인천 61.4%(2단계) 82.3%(3단계), 수원 78.2%(2단계) 92.9%(3단계), 안산 93.4%(2단계) 99.7%(3단계)등으로 나타나, 인구밀도가 균등하다고 가정하면 서울의 경우 91.2%(2단계)의 시민이 500m내에, 그리고 99.7%(3단계)의 시민이 700m내에 오픈스페이스를 가지고 있어 각각 도보로 8분, 11분이내에 도달, 이용 가능한 것으로 해석할 수 있다.

Ⅲ. 결 론

近來에 들어 더욱 深化된 都市化현상은 도시내 오픈스페이스 면적의 절대적인 부족을 가져왔고 더우기 餘暇善用에 대한 도시민의 意識變化와 이에 따른 오픈 스페이스에 대한 상대적인 需要의 증가는 도시내 오픈 스페이스의 기능상 効用가치를 확대시켜왔다.

따라서 도시 오픈스페이스의 양적인 확보 및 시설과 환경의 질적인 향상은 당면한 시급한 문제라 할 수 있으며 특히 이용의 效用性 및 均等性이란 측면에서 도시민의 오픈스페이스에 대한 접근성 문제는 매우 중요하게 인식되고 있다.

본 고에서는 이러한 觀點에서 기존의 접근성 측정에 관한 연구를 토대로 하여 오픈스페이스에 대한 접근성 측정방법을 提言하는 한편 5개 도시를 대상지로 선정하여 실제로 측정방법을 적용하여 보았다.

측정결과 도시내 오픈스페이스 면적 비율의 경우 부산, 수원, 안산이 60% 이상을 기록하여 높은 數値를 나타냈으며 서울과 인천은 각각 45.5%, 40.0%로 나타나 5개 도시중 도시내 오픈스페이스 면적 비율이 상대적으로 낮게 평가 되었다.

또한 각 段階別 접근성이 있다고 판단된 격자수의 比率에 있어서도 단계가 높아짐에 따라 줄어는 들고

있으나 도시간에 隔差를 보여주고 있다.

특히 안산은 모든 단계에서 가장 높은 비율을 보여 신설되는 계획도시의 경우 오픈스페이스에 대한 접근성의 고려가 기존도시 보다는 체계적으로 이루어지고 있음을 알 수 있으며 大都市인 서울, 부산, 인천과 中小都市인 수원, 안산을 비교하여 보면 비교적 중소도시의 비율이 높아 대도시 보다는 오픈스페이스로의 접근이 容易한 것으로 예상된다.

본 고에서 제시한 접근성 측정 방법이 內包하고 있는 潛在力과 限界点을 요약하면 다음과 같다.

방법상의 잠재력으로 첫재 micro-computer를 이용함으로써 지방 중소 행정 기관에서도 效率的이고 經濟的인 분석이 가능하며, 둘째 巨視的인 분석 기법으로서 도시간 또는 도시내 지역간 비교가 가능하게 하여 신도시개발이나 도시재정비계획, 도시재개발계획 시 중요한 指針을 제공할 수 있으리라 예상된다.

이러한 잠재력에 반하여 한계점으로 지적될 수 있는 것은 첫째 거시적(Macro한)분석인 만큼 도시계획도상의 지역구분을 기준으로 하기 때문에 정확도가 떨어지는 경향이 있으며, 둘째 접근성의 변수로 직선거리를 채택하여 도로 및 지형 등의 접근성 障碍要因에 대한 고려가 미흡하다.

그러나 이러한 분석 기법은 도시와 도시 사이의 도시 오픈스페이스에의 시민들의 접근성의 개요를 밝혀주므로 어느 도시에 오픈스페이스 확대공급의 우선 순위가 주어져야 하는 가를 알려주어 政策樹立과 投資配分의 지침을 제공하여 주며, 더욱 중요한 의의는 그러한 정책결정의 근거로서 일인당 공원면적과 같은 양적인 자료가 아니라 실제로 시민들이 쉽게 접근하여 이용하기에 용이한 오픈스페이스가 얼마나 있는가 하는 자료에 근거한다는 점이다. 또한 한 도시내에서 區別로 이와같은 분석을 시행하여 어떤 區에, 어떤 위치에 새로이 오픈스페이스를 마련해야 할 것인가를 결정하는 대안평가 수단과 정책수립의 근거를 제공하여 준다.

Ⅳ. 인용문헌

1) D.R.Ingram (1971) "The Concept of Accessibility : A Search for an Operational Form", *Regional Studies*, Pergamon Press, 5 : 101－102
2) Kevin Lynch (1971) *Site Planning*, MIT Press, : 352－353
3) R.S.Baxter (1974) "The Measurement of Relative Accessibility," *Regional Studies*, 9 : 16－17
4) 서울특별시 (1985) 서울시 공원녹지 정책방향 연구 :4
5) 윤정섭 (1985) 신편도시계획, 문운당 :147
6) 이일병, 김홍진 (1985) "서울시 도시공원에 관한 연구", 대한국토계획 학회지 국토계획, 20(1):112
7) 임강원 (1986) 도시교통계획－이론과 모형, 서울대학교 출판부:390－397
8) 황기원외 (1989) "도시공원녹지계획", 조경계획론, 문운당:127
9) 양윤재 (1985) "주거환경 설계 방법론", 환경논총, 서울대학교 환경대학원 17:121

〈부 록〉

```
/*******************************************************/
/*                                                     */
/*      ACCESS.C ; This program calculates             */
/*                 the accessibility of the            */
/*                 open space.                         */
/*                                                     */
/*      Programmer : Cho, Hyung Jun                    */
/*                                                     */
/*      Language   : IBM/PC                            */
/*                   Turbo-C Ver. 2.0                  */
/*                                                     */
/*      Last Updated Date : 1990. 8. 5.                */
/*                                                     */
/*******************************************************/

#include <stdio.h>
#include <dos.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
#include <graphics.h>

#define BORDER 1
#define ESC 27
#define REV_VID 0x70
#define NORM_VID 7
#define TRUE     1
#define FALSE    0

#define ZERO     0
#define ONE      1
#define TWO      2
#define THREE    3

#define PRINTER_COL_WIDTH  80

char  **Data;
int   Row = 0, Col = 0;
int   Loadflag = FALSE;
int   Area[4];

void GetResponse(int *);
void ProcessResponse(int);
void Edit (void);
void AccessCount(void);
int Accessible(int,int,int);
void LoadFile(void);
void AllCounting(void);
void DrawMenu(void);
void EditFile(void);
void InitializeScreen(void);
void printm (int, char*);
void ClearM (int);
void GotoXY (int, int);
void DrawBorder(int,int,int,int,int,int,int,int,int,int);
void nrerror(char error_text[]);
char **cmatrix(int,int,int,int);
void free_cmatrix(char **,int,int,int,int);
void PrintFile(void);
void PrintBlock(int,int,int,int);
void GraphFile(void);

void main(void)
{
  int choice;

  choice = 0;
  while(choice != 7){
        InitializeScreen();
        DrawMenu();
        GetResponse(&choice);
    ProcessResponse(choice);
  } /*** end of while ***/

} /******* end of main *******/

void GraphFile(void)
{
        int i,j;
        int xasp,yasp;
        int xsize,ysize;
        int graphdriver,graphmode;

        graphdriver=DETECT;
        initgraph(&graphdriver,&graphmode,"");
        getaspectratio(&xasp,&yasp);
        xsize=getmaxx();
        ysize=getmaxy();
        xsize=(xsize-Col)/2;
        ysize=(ysize-Row*xasp/yasp)/2;
        for (i=0; i<Row; i++)
          for (j=0; j<Col; j++)
                if (Data[i][j] != 0) putpixel(xsize+j,ysize
                +(int)((double)i*xasp/yasp),1);
        getch();
        closegraph();
/*       */
}

void GetResponse(choice)
int *choice;
{
  *choice = 1;

  ClearM(1);
  printm(1,"Select one item : ");
  fscanf(stdin,"%d",choice);
  ClearM(2);
  printm(2," Your choice = ");
  printf("%d",*choice);


} /***** end of GetResponse *****/

void ProcessResponse(choice)
int choice;
{
    switch(choice) {
      case 1 : printm(1,"choice = 1 Load file");
               LoadFile();
               break;
      case 2 : printm(1,"choice = 2 Edit file");
               EditFile();
               break;
      case 3 : printm(1,"choice = 3 Access count");
               AccessCount();
               break;
      case 4 : printm(1,"choice = 4 Print File");
               PrintFile();
               break;
      case 5 : printm(1,"choice = 5 All counting");
```

```
            AllCounting();
            break;
             case 6 : printm(1,"choice = 6 GraphFile");
                      GraphFile();
            break;
      case 7 : printm(1,"choice = 7 Quit program     ");
            break;
      default : printm(1,"illegal choice");
    } /*** end of switch ***/
} /***** end of ProcessResponse *****/

void EditFile(void)
{
  int  x,y;
  int  i,j;
  int  new_value;
  int  modifyflag;
  char yes;
  FILE *out_stream;
  char output_file_name[40];

  if (Loadflag == FALSE) {
    printm(2," No data is available! Please load a data file first.");
    return;
  }

  modifyflag = FALSE;
  while(1){
    ClearM(1);
    printm(1," Do you want to change a cell value (y/n) : ");
    yes = getchar();
    yes = getchar();
    if ((yes == 'n') || (yes == 'N')) break;
    ClearM(1);
    printm(1," Enter the cell position x : ");
    fscanf(stdin,"%d",&x);
         ClearM(1);
    printm(1," Enter the cell position y : ");
    fscanf(stdin,"%d",&y);
    if ((x < 0) || (x > Row) || (y < 0) || (y > Col))
      printm(2,"  Invalid position entered!!!");
    else{
      ClearM(2);
      printm(2," Enter the new_value : ");
      scanf("%d",&new_value);
      if (Data[x][y] != new_value) {
        Data[x][y] = new_value;
        modifyflag = TRUE;
      }
      GotoXY(21,3);
      printf("  Data[%d][%d] = %d          ",x,y,Data[x][y]);
    }
} /*** end of while ***/

if (modifyflag == TRUE) {
  ClearM(1);
  printm(1," Do you want to save the modified data (y/n) : ");
  yes = getchar();
  yes = getchar();
  if ((yes == 'n') || (yes == 'N')) return;

  ClearM(1);
  printm(1,"Enter output file name : ");
  scanf("%s",output_file_name);
  if ((out_stream = fopen(output_file_name,"w")) == NULL){
    printm(2," output file not opened !!!");
    }
    else{
      for (i = 0; i < Row; i++)
        for (j = 0; j < Col; j++)
          fprintf(out_stream,"%d\n",Data[i][j]);
    }
  }

} /***** end of EditFile() *****/

void AccessCount()
{
  int   i,j,n;
  int   count;
  float accessibility;

  if (Loadflag == FALSE) {
    printm(2," No data is available! Please load a data file first.");
    return;
  }

  ClearM(1);
  printm(1,"Enter No. of cell to be considered : ");
  scanf("%d",&n);

  count = 0;
  Area[ONE] = 0;

  for (i = 0; i < Row; i++)
    for (j = 0; j < Col; j++)
      if (Data[i][j] == ONE) {
        Area[ONE] += 1;
        if (Accessible(i,j,n) == TRUE) count++;
      }
  accessibility = ((float) count) / ((float) Area[ONE]) * 100.0;

  ClearM(2);
  GotoXY(21,3);
  printf(" No. of accessibility cell = %d (%4.1f\%), ",count,accessibility);
  printf("All 1 cell = %d",Area[ONE]);

} /***** end of AccessCount() *****/

int Accessible(x,y,n)
int x;
int y;
int n;
{
  int i,j;
  int tx,ty;
  int dist;

  for (i = -n; i <= n; i++)
    for (j = -n; j <= n; j++){
      tx = x + i;
      ty = y + j;
        dist= ceil(sqrt( (double)i*i + (double)j*j)) <= n;
        if (dist && (tx >= 0) && (tx < Row) && (ty >= 0)
         && (ty < Col)) {
                        if (Data[tx][ty] == TWO) return(TRUE);
      }
    }
  return(FALSE);

} /***** end of Accessible *****/
```

```
void AllCounting()
{
  int i,j;
  float sum,percent;

  if (Loadflag == FALSE) {
    printm(2," No data is available! Please load a data file first.");
    return;
  }

  for (i = 0; i < 4; i++)
    Area[i] = 0;

  for (i = 0; i < Row; i++)
    for (j = 0; j< Col; j++)
      Area[Data[i][j]] += 1;

  sum = 0.0;
  for (i = 1; i < 4; i++)
    sum += Area[i];

  GotoXY(21,3);
  for (i = 1; i < 4; i++){
    percent = Area[i] / sum * 100.0;
    printf("Area[%d] = %d (%4.1f\%), ",i,Area[i],percent);
  }

} /***** end of AllCounting() *****/

void LoadFile()
{
  char input_file_name[20];
  FILE *in_stream;
  char **cmatrix();
  void free_cmatrix();

  int  i,j;
  char in_data;

  printm(1,"Enter input file name : ");
  scanf("%s",input_file_name);
  if ((in_stream = fopen(input_file_name,"rt")) == NULL){
    printm(2," input file not found !!!");
  }
  else{
    if (Loadflag == TRUE) {
      printm(2,"Freeing old matrix...                    ");
      free_cmatrix(Data,0,Row,0,Col);
    }
    printm(1,"                                                    ");
    printm(1,"Enter the No. of row : ");
    scanf("%d",&Row);
    printm(1,"                                                    ");
    printm(1,"Enter the No. of column : ");
    scanf("%d",&Col);
/*
    printf("row = %d, col = %d",Row,Col);
*/
    Data = cmatrix(0,Row,0,Col);
    printm(2," Now reading input file ...");
    for (i = 0; i < Row; i++)
      for (j = 0; j < Col; j++){
        fscanf(in_stream," %c ",&in_data);
        Data[i][j] = atoi(&in_data);
      }
/*
    for (i = 0; i < Row; i++){
      for (j = 0; j < Col; j++){
        printf(" %d",Data[i][j]);
      }
      printf("\n");
    }
*/
    Loadflag = TRUE;
  } /*** end of else ***/

} /***** end of LoadFile() *****/

void  DrawMenu()
{
  DrawBorder(7,30,15,50,196,179,218,191,192,217);
  GotoXY(8,32);
  printf("1. Load file");
  GotoXY(9,32);
  printf("2. Edit file");
  GotoXY(10,32);
  printf("3. Access count");
  GotoXY(11,32);
  printf("4. Print file");
  GotoXY(12,32);
  printf("5. All counting");
  GotoXY(13,32);
  printf("6. Plotting data");
  GotoXY(14,32);
  printf("7. Quit program");

} /***** end of DrawMenu *****/

void InitializeScreen()
{
  system("cls");
  DrawBorder(0,0,23,79,196,179,218,191,192,217);
  DrawBorder(3,25,5,55,205,186,201,187,200,188);
  GotoXY(4,34);
  printf("Accessibility");
  DrawBorder(19,1,22,78,196,179,218,191,192,217);
  GotoXY(19,2);
  printf("Message Box");

} /***** end of InitializeScreen *****/

void printm(line,message)
int line;
char *message;
{
  GotoXY(19+line,3);
  printf(message);

} /***** end of printm *****/

void ClearM(line)
int line;
{
  GotoXY(19+line,3);
  printf("
}

void GotoXY(x,y)   /* send the cursor to x,y */
int x,y;
{
  union REGS r;
```

```
  r.h.ah = 2;
  r.h.dl = y;
  r.h.dh = x;
  r.h.bh = 0;
  int86(0x10,&r,&r);
} /***** end of GotoXY *****/

void DrawBorder(startx,starty,endx,endy,hl,vl,ulc,urc,llc,lrc)
int startx,starty,endx,endy;
int hl,vl,ulc,urc,llc,lrc;
{
  register int i;

  for(i = startx+1; i < endx; i++){
    GotoXY(i,starty);
    putchar(vl);
    GotoXY(i,endy);
    putchar(vl);
  }

  for(i = starty+1; i < endy; i++){
    GotoXY(startx,i);
    putchar(hl);
    GotoXY(endx,i);
    putchar(hl);
  }

  GotoXY(startx,starty);
  putchar(ulc);
  GotoXY(startx,endy);
  putchar(urc);
  GotoXY(endx,starty);
  putchar(llc);
  GotoXY(endx,endy);
  putchar(lrc);

} /***** end of DrawBorder *****/

/*****************************************************************
        Memory management utilities
*****************************************************************/

void nrerror(error_text)
char error_text[];
/* standard error handler */
{
  void exit();

  printm(1,"Access run-time error ...");
  printm(2,error_text);
  exit(1);
} /***** end of nrerror() *****/

char **cmatrix(nrl, nrh, ncl, nch)
int nrl, nrh, ncl, nch;
/* Allocates an char matrix with range [nrl..nrh][ncl..nch] */
{
  int  i;
  char **m;

  /* allocate pointers to rows */
  m = (char **)malloc((unsigned) (nrh - nrl + 1) * sizeof(char*));
  if (!m) nrerror("allocation failure 1 in cmatrix()");
  m -= nrl;
/*
printf(" pointers to rows are allocated\n");
*/
  /* allocate rows and array of pointers to them */
  for (i = nrl; i <=nrh; i++){
    m[i] = (char *)malloc((unsigned) (nch - ncl + 1) * sizeof(char));
    if (!m[i]){
      nrerror("allocation failure 2 in cmatrix()");
      printf("i = %d\n",i);
    }
    m[i] -= ncl;
  }

  /* return pointer to array of pointers to rows */
  return m;
} /***** end of imatrix() *****/

void free_cmatrix(m, nrl, nrh, ncl, nch)
char **m;
int nrl,nrh,ncl;
/* Frees a matrix allocated with cmatrix */
{
  int i;

  for (i = nrh; i >= nrl; i--) {
    free((char*) (m[i]+ncl));
/*
    printf("in free_cmatrix i = %d\n",i);
*/
  }
  free((char*) (m+nrl));
} /***** end of free_imatrix() *****/


void PrintFile()
{
  int  i,j;
  int  ccount;
  int   startRow, startCol;   /* used for print file */
  int   lastRow, lastCol;
  char lf[4] = { 27,'3',17,0 };
  char lf1[3] = {27,'2',0} ;
  ccount = Col / PRINTER_COL_WIDTH;
  startRow = 0;
  startCol = 0;
  lastRow = 0;
  lastCol = 0;

  fprintf( stdprn,"%s",lf);

  for (i=0; i<ccount; i++){
    startCol = i * PRINTER_COL_WIDTH;
    lastCol  = startCol + PRINTER_COL_WIDTH;
    lastRow  = startRow + Row;
    printm(2,"");
    printf("printing from (%d,%d) to (%d,%d)",startRow,startCol,lastRow,
    lastCol);
    PrintBlock(startRow,startCol,lastRow,lastCol);
   } /*** end of for (i=0; i<ccount; i++) ***/

  if (lastCol < Col){
    startCol = lastCol;
    lastCol  = Col;
    lastRow  = startRow + Row;
    printm(2,"");
    printf("printing from (%d,%d) to (%d,%d)",startRow,startCol,lastRow,
    lastCol);
    PrintBlock(startRow,startCol,lastRow,lastCol);
   } /*** end of if (lastCol < Col) ***/
  fprintf(stdprn,"%s",lf1);

} /***** end of PrintFile *****/

void PrintBlock(sr,sc,lr,lc)
int sr, sc, lr, lc;
{
  int  i,j;

  for (i=sr; i<lr; i++){
    for (j=sc; j<lc; j++){
                if (Data[i][j] == ZERO) fprintf(stdprn," ");
                if (Data[i][j] == ONE)   fprintf(stdprn,"L");
      if (Data[i][j] == TWO)   fprintf(stdprn,"O");
      if (Data[i][j] == THREE) fprintf(stdprn,"X");
    }
    fprintf(stdprn,"\n");
  }
  fputc(12,stdprn);
/*
  fprintf(stdprn,"\12");
*/
} /***** end of PrintBlock() *****/
```