```
//dlugosc char 5.2.2
int dlug1(int n,char tekst[]){
    int i=0;
    while(tekst[i]!=0){
          i++;
    }
    return i;
}

//dlugosc wchar 5.2.2
int dlug2(int n,wchar_t tekst2[]){
    int i=0;
    while(tekst2[i]!=0){
          i++;
    }
    return i;
}

//wywolania main char i wchar
    char tekst[20]="przechowywany";
    wchar_t tekst2[20]=L"napisy";

    printf("Dlugosc napisu: %d\n",dlug1(n,tekst));
    wprintf(L"Dlugosc napisu: %d",dlug2(10,tekst2));

//sprawdzanie ktory napis jest wczesniej w kolejnosci w
alfabecie 5.2.4
int spr(char nap1[],char nap2[]){
    for(int i=0;i<10;i++){
        if(nap1[i]<nap2[i])
            return 1;
        else return 0;
    }
}

//kopiuje n pierwszych znakow z jednej tab do drugiej   5.2.6
void kopiujn(int n, char nap1[],char nap2[]){
    for(int i=0;i<n;i++){
        nap2[i]=nap1[i];
    }
    nap2[n]='\0';
}
void kopiujn2(int n, wchar_t nap1w[],wchar_t nap2w[]){
    for(int i=0;i<n;i++){
        nap2w[i]=nap1w[i];
    }
    nap2w[n]='\0';
}

//wycinanie pierwszego wystapienia 5.2.10
void wytnij2(char nap1[], char nap2[],char nap3[])
{
    for(int i=0;nap1[i]!=0;i++){
        for(int j=0; nap2[j]!=0;j++){
            if (nap2[j]!=nap1[i+1]&&nap2[j]!=nap1[i+2]&&nap2[j
]!=nap1[i+3]);
            nap3[i]=nap1[j];
        }
    }
}
//trzy napisy i zwraca jako wartoœæ napis powsta³y ze sklejenia
napisów 5.2.22
int *sklej(char *nap1, char *nap2, char *nap3){

    char *wyn=malloc((strlen(nap1)+ strlen( nap2 )+ strlen( nap3
)+1) * sizeof(char));
```

```
    strcpy(wyn,nap1);
    strcat(wyn,nap2);
    strcat(wyn,nap3);

    return wyn;
}

//dynamiczn¹ dwuwymiarowa tablica tablic elementów typu int o
wymiarach n na m, i zwraca jako wartoœæ wskaŸnik do niej 6.2.1
int **dyna(int n,int m){
    int**tab=(int**) malloc(sizeof(int*)*n);
    for(int i=0;i<n;i++){
        *(tab+i)=malloc(sizeof(int)*m);
    }
    return tab;
}

//usuwa z pamieci tablice 6.2.5
void usun(int n,int m,int **tab){
    for(int i=0;i<n;i++){
        free(*(tab+i));
    }
    free(tab);
}

//dynamiczn¹ trojwymiarowa tablica tablic 6.2.5
int **dyna(int n,int m){
    int**tab=(int**) malloc(sizeof(int*)*n);
    for(int i=0;i<n;i++){
        *(tab+i)=malloc(sizeof(int)*m);
    }
    return tab;
}

//dynamiczna dwuwymiarowa trojkatna tablica tablic 6.2.7
int **troj(int n){
    int**tab=(int**) malloc(sizeof(int*)*n);
    for(int i=0;i<n;i++){
        *(tab+i)=malloc(sizeof(int)*(i+1));
    }
    return tab;
}

//suma wartosci elementow tablicy 2wymiarowej 6.2.11
int sumdw(int tab[][100],int n){
    int S=0;
    for(int i=0;i<n;i++){
        for(int j=0;j<100;j++){
            tab[i][j]=S*tab[i][j];
        }
    }
    return S;
}

//suma wartosci elementow tablicy 3wymiarowej 6.2.14
int sumdw(int tab[][100][100]){
    int S=0;
    for(int i=0;i<100;i++){
        for(int j=0;j<100;j++){
            for(int k=0;k<100;k++){
                    S=S+tab[i][j][k];
            }
        }
    }
    return S;
}
```

```
//najwiêksz¹ spoœród œrednich wartoœci elementów poszczególnych wierszy 6.2.17
int sred(int **tab,int n, int m){
    int max=0;
    for(int i=0;i<n;i++){
        int srednia=0;
        for(int j=0;j<m;j++){
            srednia+=tab[i][j];
        }
        if((srednia/m)>max)
            max=(srednia/m);
    }

    return max;

}
int main()
{
    int**tab= (int**)malloc(sizeof(int*)*2);
    tab[0]=(int*)malloc(sizeof(int)*2);
    tab[1]=(int*)malloc(sizeof(int)*2);
    *(*(tab+0)+0)=10;
    *(*(tab+0)+1)=8;
    *(*(tab+1)+0)=2;
    *(*(tab+1)+1)=4;
    printf("MAX= %d",sred(tab,2,2));
    return 0;
}

//odwraca kolejnoœæ elementów we wszystkich wierszach otrzymanej tablicy 6.2.22
void obracanie(int n, int m, int tab[][m]){
    int temp=0;
    for(int i=0;i<n;i++){
        temp=tab[i][0];
        for(int j=0; j<m; j++){
            tab[i][j]=tab[i][m-j-1];
        }
        tab[i][m-1]=temp;
    }
}

//wypisywanie tablicy 2wymiarowej
void pisz(int n, int m, int tab[][m]){

    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            printf("%d ",tab[i][j]);
        }
        printf("\n");
    }
}

//zmienia kolejnoœæ kolumn w tablicy w taki sposób, ¿e kolumna pierwsza ma siê znaleŸæ na miejscu drugiej  6.2.24
void obracanie(int n, int m, int **tab){
    int temp;
    for(int i=0; i<n; i++)
    {
        temp = tab[i][m-1];
        for(int j=m; j>=0; j--)
        {
            tab[i][j] = tab[i][j-1];
        }
        tab[i][0] = temp;
    }
```

```
}

//obwod trojkata struktura 7.2.1
struct trojkat{
    int a;
    int b;
    int c;
};

int obw(struct trojkat boki){
    return boki.a+boki.b+boki.c;
}

//przepisuje zawartosc z jednej do drugiej zmiennej 7.2.2
struct trojkat{
    int a;
    int b;
    int c;
};
void przep(struct trojkat troj1, struct trojkat *troj2){
*troj2=troj1;
}

//przepisuje zwartoœæ tablicy tab1 do tablicy tab2 7.2.6
struct punktn{
    double *a;
    int b;
};

void przep(struct punktn tab1[], struct punktn tab2[], int n){
    for(int i=0;i<n;i++){
        tab2[i].a=malloc(tab1[i].b*sizeof(double));
        for(int j=0;j<tab1[i].b;j++){
            tab2[i].a[j]=tab1[i].a[j];
        }
    }
}

//zwraca sume dwoch argumentow 7.2.7
struct zespolone{
    double im;
    double re;
};

double dodaj(struct zespolone zb1,struct zespolone zb2){
    return zb1.im+zb1.re+zb2.im+zb2.re;
}

//struktura dwa pola jedno int a drugie wskaznik do tego typu
7.2.9
struct lista{
    int pole;
    struct lista*wsk;
};

//bêdzie mo¿na przechowywaæ zarówno zmienne typu int, jak i
unsigned int 7.2.10
union super_int{
    int a;
    unsigned int b;
};

// 7.2.11
union Liczba{
    int a;
    float b;
```

```
};
struct Dane{
    int tp;
    union Liczba zaw;
};
struct Dane wczytaj(){
    struct Dane temp;
    printf("Jesli chcesz liczbe calk podaj 0, jesli wym to 1\n"
);
    scanf("%d",&temp.tp);
    if(temp.tp==0){
        printf("Poda liczbe calk\n");
        scanf("%d",&temp.zaw.a);
    }
    else if(temp.tp==1){
        printf("Poda liczbe wym\n");
        scanf("%f",&temp.zaw.b);
    }
    return temp;
};
void pokaz(struct Dane l){
    if(l.tp==0){
        printf("Podana liczba: %d",l.zaw.a);
    }
    else if(l.tp==0){
        printf("Podana liczba: %f",l.zaw.b);
    }
}

//przyjmowaæ wartoœci odpowiadaj¹ce nazwom ró¿nych zwierz¹t
domowych 7.2.15
enum zwierzak{
    pies,
    kot,
    chomik,
    rybki,
    krolik,
};

//dodawanie na poczatek listy 7.3.3
struct element{
    int i;
    struct element*next;
};
struct element*tworz(){
    return NULL;
};

struct element*dodaj(struct element*Lista, int a){
    struct element*wsk=malloc(sizeof(struct element));
    wsk->i=a;
    wsk->next=Lista;
    return wsk;

}

void wyswietlLBG(struct element*Lista){
    struct element*temp=Lista;
    if(temp==NULL){
        printf("Lista jest pusta\n");
    }
    while(temp!=NULL){
        printf("%d\n",temp->i);
        temp=temp->next;
    }
    printf("----\n");
```

```
}
//
//
```