

Wprowadzenie do testowania			
Przyczyna Błąd – usterka (defekt, pomyłka, rozbieżność) w programie, która może skutkować nieprawidłowym (tj. niezgodnym z oczektwaniami użytkownika) działaniem Skutkiem błędu w programie może wykonań Takie samo błędne wykonanie progranych błedów	, , , , , , ,		
Cotosiet - najleptaz investycja Projekt nr PCKL 0.4.01 01 01 02 11 10 / Aztalionnie dia grapod sepolifinatorewsky przez Ulię Europojdką se art. "Cacholin Europojd sepolifinatorewsky przez Ulię Europojdką se art. Sepolifinatorewsky pr	anki opantej sa wiedzy" dolego Funduszi Spułecznego www.nacentoka. Fancor (1960)		

(A)

Wesley Professional, 2003

Człowiek – najlepsza inwestycja







Wprowadzenie do testowania □ Najbardziej spektakularne błędy w programach oraz ich skutki: • 1988: Zestrzelenie samolotu Airbus A320 - śmierć 290 osób • 1991: Błąd naprowadzania rakiet Patriot - śmierć 28 osób • 1994: Błąd dzielenia zmiennoprzecinkowego w procesorze Pentium - koszt ok. 475 000 000 USD • 1996: Katastrofa rakiety Ariane 5 - koszt ok. 500 000 000 USD • 1999: Katastrofa rakiety Ariane 5 - koszt ok. 500 000 000 USD • 1999: Katastrofa sondy kosmicznej Mars Polar Lander 5 - koszt samego lądownika ok. 120 000 000 USD • 2000: "pluskwa milenijna"

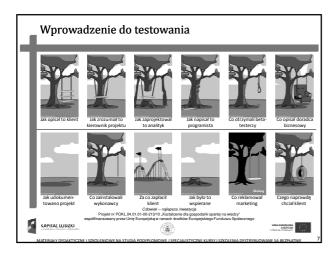
Wprowadzenie do testowania	
☐ Kiedy w oprogramowaniu znajduje się błąd? • Oprogramowanie nie wykonuje czegoś, co według specyfikacji	
powinno wykonywać	
 Oprogramowanie robi coś, czego według specyfikacji nie powii robić 	nno
 Oprogramowanie wykonuje coś, o czym specyfikacja w ogóle n wspomina 	iie
 Oprogramowanie nie wykonuje czegoś, o czym specyfikacja nie wspomina, ale powinna 	е
 Oprogramowanie jest trudne do zrozumienia, trudne do użycia powolne albo – zdaniem testera – będzie w oczach użytkownił prostu nieprawidłowe' 	
*źródło: R. Patton, <i>Testowanie oprograma</i>	owania
Cokowiek – najlipsza inwestycja Projekt nr POKL-04.01.01-00.213/10 "Kazalicanie dla gospodarki opartej na wiedzy" współlinansowany przez Unię Europejska w ramach srodków Europejskogo-Funduszu Społecznego	
KAPITA: LUDZKI METAGORI STATIGLI PROMISO FRANCE POLICE FRANCE POLICE METAGORI STATIGLI PROMISO FRANCE POLICE FRANCE	

Wprowadzenie do testowa	ania
☐ Źródła powstawania błędóv	v:
	■ Sptcification ■ Design ■ Code ■ Other źródło: R. Patton, <i>Testawanie oprogramowania</i>
☐ Istotnymi czynnikami mając błędów w programach jest i	cymi wpływ na powstawanie ch rosnąca złożoność i wielkość
Projekt nr POKL 04.01.01.00.21910 . Kaza wapolitinarsowany przez Unię Europejską w ranko wapolitinarsowany przez Unię Europejską w ranko w ran	eltonie dla gespodarki opartej na wiedzy* środków Europejskiego Funduszu Społecznego una successa Facesz Poucze

Człowiek – najlepsza inwestycja







Wprowadzenie do testowania	
■ Błędy w programie można rozpatrywać na poziomie: • mechanizmu powstawania błędu (error), związanego najczęściej z fazą analizy wymagań i projektowania (błędne założenia projektowe, niewłaściwa interpretacja specyfikacji, dobór niewłaściwego algorytmu, itp.) • programowym (fault), związanym najczęściej z fazą implementacji (błędy programisty) • użytkowym (failure), kiedy można zaobserwować błędne działanie funkcji programu	
☐ Dynamiczny rozwój branży IT powoduje obserwowalne i rosnące koszty będące skutkami błędów w programach Catowie - najkraza inwestycja Projekt or PONL 0.0.10 1-00 20 2010 W działowie dą opspodowi opanią na wiedzy'	
wapdifinansowany przez Unije Europejak w ramach śrotkie Europejakiego Fundustu Społeczmego APITAL LUDZKI WOMA Engografi APITAL LUDZKI WOMA ENGORAT APITAL LU	,

Wprowadzenie do testowania				
☐ Szacu	☐ Szacunkowe koszty błędów:			
	Faza	Koszt [€]	Czas naprawy [h]	
	Analiza	1	0.1	
	Projektowanie	5	0.5	
	Kodowanie	10	1	
	Testowanie	15-50	1.5-5	
	Wdrożenie	>100	>10	
		Człowiek – naiteosza inwest	annia.	
KAPITAŁ LUDZKI MARODOMA SPACEGIA SPORM	współfinansowany przez Unię E	01-00-213/10 "Kształcenie dla	tycja gospodarki opartej na wiedzy" Europejskiego Funduszu Społecznego	UNIA EUROPUSAA EUROPUSA FIADUSE POLICOY







_ Wprowadzenie do testowania	
☐ Czy testowanie się opłaca?	
Scenariusz #1: Lighb blodów malarianych w programia 1000	
o Liczba błędów znalezionych w programie − 1000 o Liczba błędów znalezionych przez deweloperów − 750	
o Liczba błędów znalezionych przez użytkowników – 250 o Średni koszt wykrycia i naprawy błędu: przed wdrożeniem – 10€,	
po wdrożeniu – 250€ ⊙ Koszt zapewnienia jakości:	
750 * 10€ + 250 * 250€ = 70 000€	
Colosiel - najlepuza Investivcja	
Pojede nr POXL 0x 01 0.1 00:213 PL Sustitionis da propodark (oparte) na viector) wepdiffinancewary przez Linię Europojską w amach berodow Europojskiego Funduzus Społecznego CAPITAL LUDZKI ***********************************	
MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALISTYCZNE RUBSY I SZKOLENIA DYSTRYBUOWANE SA BEZPŁATNIE 10	
_ Wprowadzenie do testowania	
 Scenariusz #2: Liczba błędów znalezionych w programie – 1000 	
 Liczba błędów znalezionych przez deweloperów – 750 Liczba błędów znalezionych przez testerów – 200 	
o Liczba błędów znalezionych przez użytkowników – 50 o Średni koszt wykrycia i naprawy błędu: przed testowaniem – 10€,	
w fazie testowania – 50€, po wdrożeniu – 250€ ⊙ Koszt przygotowania i przeprowadzenia testów – 10 000€	
 Koszt zapewnienia jakości: 750 * 10€ + 200 * 50€ + 50 * 250€ + 10 000€ = 40 000€ 	
Wniosek: Naprawa błędnego oprogramowania jest droższa	-
(finansowo i marketingowo) niż koszt pracy testerów	
Projekt nr POKL Okt 0.10 1-00 23/19 / Kattlende da gospodaník opantej na viedný vepdífinansowany przez Unię Europojską v ramach brodnów Europojskogo Funduszu Społecznego KAPITAL LUDZKI valonim zalona prvedo Parce produ	
MATERIANY DYDACTYCZNE I SZYDI ENIOWE NA STILDIA PODWOŁOMOWE I SPCCIALISTYCZNE KURSY I SZYDI ENIA DYSTRYBUDWANE SA BEZPŁATNIE 11	
_ Wprowadzenie do testowania	
-	
☐ Testowanie oprogramowania to wykonanie kodu dla różnych	
kombinacji danych wejściowych i stanów, mające na celu wykrycie błędów w programie	
 Zasadniczym celem testowania jest dążenie do uzyskania odpowiednio wysokiego poziomu niezawodności i jakości 	
oprogramowania, stosownie do zdefiniowanych wymagań, ograniczeń czasowych i kryteriów ekonomicznych	







	1
_ Wprowadzenie do testowania	
•	
☐ Testowanie powinno umożliwiać wykrycie jak najwiekszej	
liczby błędów → wzrost niezawodności programu	
☐ Testowanie powinno zapewniać na jak najwcześniejsze	
wykrycie błędów → redukcja kosztów naprawy błędów	
1	
☐ Test zakończony sukcesem to test, którego skutkiem jest	
wykrycie nieznalezionego dotychczas błędu w programie	
☐ Test uznaje się za efektywny, jeżeli ma zdolność wykrycia	
jak największej liczby błędów z jak największym	
prawdopodobieństwem	
Człowiek – najłopsza inwestycja Projekt nr POKL 04. 0.1 0-10-0 21310 (Asztalsonie dla gospodanki opantej na wiedzy* współfinansowany przez Uhię Europejak w armach stodziów zuczpejakkago Funduszu Społecznego	
wsportmans/wany przez Unię Europajską w ramach srookow Europajskago i Funduszu Sporecznego KAPITAL LUDZKI SIGNOFICA FINANCIA PORTOR FINANCIA FINANCIA PORTOR FINANCIA FINANCIA PORTOR FINANCIA FINANCIA PORTOR FINANCIA FINANCIA PORTOR FINANCIA PORTOR	
MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA BEZPŁATNIE 13	
*** 1 1 1	
Wprowadzenie do testowania	
Uruchamianie programów ("odpluskwianie", debugowanie)	
jest działaniem mającym na celu zlokalizowanie, a następnie	
poprawienie wykrytych w trakcie testowania błędów	
Do uruchamiania programów wykorzystuje się przypadki	
testowe zaprojektowane w taki sposób, aby były pomocne	
w znajdowaniu przyczyny błędu	
Uruchamianie programów jest procesem, w trakcie którego	
formułuje się, a następnie weryfikuje praktycznie hipotezy,	
dotyczące przyczyn błędnego działania programu	
☐ W trakcie debugowania bardzo pomocne jest doświadczenie	-
i wiedza, dotycząca typowych błędów programistycznych	
Calowiek – najlepsza inwestycja	
Projekt nr POKLO4.01.01-00.213/10. Ksztalicenie dla geptodanki opartej na wiedzy' współfinanzowany przez Unię Europejskią nr ramach szodów Europejskiego Funduszu Społecznego WARDITAL LUDZKI WARDITAL LUDZKI WARDITAL LUDZKI	
ANTORNI FINICIA PRINCIPI CINICI FINICI FINIC	
MALE RIGHT TITLBACHT ZWE S SALDE PNE DWF RA STITUTA PULIFICATION TWF S SPECIAL STATE ARREST SZADLE POLA TITSTER TO STATE	
	1
_ Wprowadzenie do testowania	
☐ W trakcie przygotowywania testów oraz w procesie samego	
testowania należy przestrzegać pewnych ogólnych zasad, np.	
■ podczas testowania ważniejsze jest sprawdzenie typowych	
przypadków użycia programu, niż przypadków rzadkich (skrajnych)	
testowanie istniejącej funkcjonalności programu powinno być	
ważniejsze od testowania nowej funkcjonalności należy unikać testowania przez autorów oprogramowania	
 naieży unikać testowania przez autorow oprogramowania należy dokumentować testy i zasady ich użycia 	
 należy dokumentować testy i zasady ich użycia należy zbierać i poddawać analizie wyniki testów 	
 trzeba uwzględniać w testach zarówno poprawne, jak i niepoprawne 	
dane wejściowe oraz warunki wykonania	
 należy sprawdzać, nie tylko czy w trakcie działania program wykonuje 	
to co powinien, ale także czy nie wykonuje tego, czego nie powinien	
Calowiek – najlepsza inwestycja Projekt nr POKL 0.4. 0.1.0-10-0.2 1310 . Katalacenie dla gospodarki opartej na wiedzy' współlinansowany przez Unię Europejsia w ramach śrośców kroupojskiego Funduszu Społecznego	







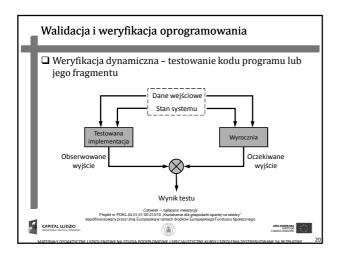
Walidacja i weryfikacja oprogramowania	
Atestowanie, walidacja (<i>validation</i>) jest sprawdzaniem zgodności działania programu z oczekiwaniami użytkownika w rzeczywistych warunkach i w środowisku, dla których został on zaprojektowany Walidacja zajmuje się wykrywaniem błędów związanych z nieprawidłową specyfikacją wymagań	
☐ Weryfikacja (<i>verification</i>) jest procesem mającym na celu sprawdzenie, czy program na danym etapie wytwarzania jest zgodny z wymaganiami i oczekiwaniami użytkownika Weryfikacja nie zajmuje się wykrywaniem błędów związanych z nieprawidłową specyfikacją wymagań	
Projekt or PORLO 94.10 - Analysis investigate investigate investigate participate of participate	
	1
Walidacja i weryfikacja oprogramowania	
☐ Walidacja stara się odpowiedzieć na pytanie: "Czy budujemy prawidłowy produkt?"	
i w tym sensie dotyczy całościowej oceny programu pod kątem spełniania oczekiwań użytkownika □ Weryfikacja natomiast stara się odpowiedzieć na pytanie:	
"Czy prawidłowo budujemy produkt?" i obejmuje szerszy zakres zagadnień niż testowanie	
☐ Zarówno weryfikacja, jak i walidacja powinny być wykonywane na każdym etapie tworzenia oprogramowania	
Projek or POKILO4 0.1 0-00 2-130 Skateland dispaposah spanje na wiedzy współnanzowany przez Unię Europejskie y raminh środnie Europejskiego Funduszu Społecznego waskiego pokinie za pod p	
	_
Walidacja i weryfikacja oprogramowania	
□ Weryfikacja może mieć charakter: • statyczny – związany z analizą (przeglądem, inspekcją) specyfikacji,	
projektu lub kodu programu (bez konieczności jego uruchamiania) ■ dynamiczny – polegający na wykonaniu programu lub jego fragmentu z danymi testowymi i porównaniu uzyskanych wyników z wynikami poprawnymi → testowanie oprogramowania	
☐ Typową techniką statycznej weryfikacji są <u>inspekcje kodu,</u> podczas których zespoły inspekcyjne dokonują przeglądu kodu pod kątem sprawdzenia jego poprawności oraz	
spełnienia dodatkowych kryteriów (prostota, poprawność zastosowanych konstrukcji programistycznych, zgodność ze standardami kodowania,)	
Colomiel - najlopaza Investrycja Projek nr POKL D4.01.01.00 213 10 "Katalsonie da gospodarki opartij na wiedzy* współlinacowany przez Unię Europejską w ramach środkie Europejskapi Funduzu Spółecznego KAPITAŁ LUDZKI	







V	Nalidacja i weryfikacja oprogramowania
T	Inną formą statycznej weryfikacji kodu jest zastosowanie narzędzi do <u>automatycznej analizy kodu</u>
ı	Tego typu analizatory przeglądają kod źródłowy analizując jego składnię, np. w celu sprawdzenia poprawności użytych konstrukcji programistycznych, czy też wnioskowania
ı	o przepływie sterowania w programie
	Automatyczne analizatory kodu bazują na metodach formalnych, które dla wielu systemów generują bardzo
	wysokie koszty, a ponadto nie są w stanie poradzić sobie
	z trudnościami podczas definiowania wymagań
A s	Calculate — najlopaza innestrycia Projekt nr POKL 0.4.01 of 10-03-110 Azialationen dia appropolatiki opasteji na wiedzy' wepdifinanzowany przez Unię Europojiką w ramachi śrotków Europojikiego Funduszu Społecznego APITAL LUDZKI
MAT	TERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBLIOWANE SA BEZPŁATNIE 19



Walidacja i weryfikacja oprogramo	owania
☐ Kod poddawany testowaniu to:	
 implementacja testowana – IUT (Implem 	antation Under Teat
 metoda testowana – MUT (Method Under 	· · · · · · · · · · · · · · · · · · ·
`	,
 obiekt testowany – OUT (Object Under Tollies) klasa testowana – CUT (Class Under Test 	·
składowa testowana – CUT (Component	,
system testowany – SUT (System Under 1	<i>'</i>
• system testowany – 501 (<i>System Under L</i>	iest)
☐ Przypadek testowy (<i>Test case</i>) – spec	cyficzny zestaw danych
wejściowych, warunków początkowy	5 5
wyników i warunków końcowych, słu	,
kodu programu	
Człowiek – naileosza inwestycia	
Projekt nr POKL.04.01.01.02.13/10, //sztalcenie dla gospodarki op współfinansowany przez Unię Europejską w ramach środków Europejskiego	
KAPITAL LUDZKI WARDOOMS TRATION SPONDSCI	UNIA EUROPEISAA RANDISZ POLICZNY
MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALISTYCZNE KURS	SY I SZKOLENIA DYSTRYBUOWANE SA BEZPŁATNIE 21







Walidacja i weryfikacja oprogramowania	
□ Typowe aktywności związane z weryfikacją: • sprawdzanie, czy wymagania użytkownika są weryfikowalne • sprawdzanie, czy elementy składowe projektu są weryfikowalne zgodnie z wymaganiami użytkownika • testowanie jednostek oprogramowania (modułów) • testowanie modułów po ich integracji • testowanie programu przez jego twórców • testowanie akceptacji programu przez użytkowników • okresowe inspekcje i przeglądy • audyt (kontrola jakości)	
C-dowek - najkopza inwestycja Projek nr POKL 04.01.01 00-02/11/01 04.catalecine da geospodarki opastnji na wiedzy' współnianowany przez Unif Europojoka y nauki Osobok Europojokago Fauduszu Spółocznego CAPITAL LUDZKI MATERIAŁY DYDAKTYCZNE I SZNOLENIOWE Na STUDIA PODVYR OMOWE I SPECIALISTYCZNE KURSY I SZNOLENIA DYSTRYBIOWANE SA BEZPIATNIE 222	
	1
Walidacja i weryfikacja oprogramowania	
☐ Testowanie jest jedyną techniką walidacji w odniesieniu do wymagań niefunkcjonalnych, takich jak np. rozszerzalność, bezpieczeństwo, wydajność czy skalowalność systemu ☐ Przeglądy wymagań (<i>reviews</i>) są najbardziej efektywną metodą walidacji wymagań	
☐ Aby osiągnąć zadowalający poziom weryfikacji i walidacji, konieczne jest stosowanie zarówno techniki statycznej weryfikacji, jak i testowania, wykorzystującego odpowiednio adekwatny zestaw przypadków testowych	
Projekt in PONL 0.6.1.0.1.0.0.2710 / Asiabere dia geododini opatria na violani na violani napodini na violani napodini na violani napodini na violani napodini napodi	
	1
Walidacja i weryfikacja oprogramowania	
 □ Ostatecznym celem weryfikacji i walidacji nie jest sprawdzenie, czy kod programu jest wolny od błędów, lecz czy będzie on spełniał swoje funkcje i nadawał się do wykorzystania przez użytkownika □ Osiągnięcie tak zdefiniowanego celu można utożsamiać z uzyskaniem odpowiedniego poziomu pewności działania programu 	
☐ Poziom pewności zależy przede wszystkim od: • przeznaczenia oprogramowania (stopnia krytyczności) • oczekiwań użytkowników • uwarunkowań na rynku Projekt n POKL 04.01 // 10 /02 23 10 // Actientem da popopoli Projekt n POKL 04.01 // 10 /02 23 10 // Actientem da popopoli na wiedzy/ repolificanciowany przez bliegi projekt w manch dode Europelpiego Fundazu Godecznogo Projekt na pokl. 04.01 // 10 /02 23 10 // Actientem da popopoli na wiedzy/ repolificanciowany przez bliegi projekt w manch dode Europelpiego Fundazu Godecznogo Projekt na pokl. 04.01 // 10 /02 23 10 // Actientem da popopoli na wiedzy/ repolificanciowany przez bliegi projekt w manch dode Europelpiego Fundazu Godecznogo Projekt na pokl. 04.01 // 10 /02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 /02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10 // 02 23 10 // Actientem da popopoli na wiedzy/ Projekt na pokl. 04.01 // 10	
współfinansowany przez Unię Europejską w ramach środków Europejskiego Funduszu Społecznego KAPITAL LUDZKI WAN EUROPEJSKA FRANCE PROJECT FR	







Model V	,	
pokazu Podsta testo reali testo systo	V definiuje podstawowe fazy testowania s uje ich zależność z fazami projektowania s wowe fazy testowania systemu: owanie modułów – testowanie najmniejszych jedn zowane bezpośrednio po zakończeniu ich implem owanie systemu – testowanie poszczególnych pod- emu jako całości, realizowane po zintegrowaniu m dowych	ostek kodu, ostek kodu, ientacji systemów oraz iodułów
jest	owanie akceptacji użytkowników – oprogramowat testowaniu przez przyszłego użytkownika lub nie: lucenta zespół testowy (testy alfa, testy beta) Projekt nr POKLOLO I 01 00 20310 Azaldonou dla pspecidani opaniej na wiedzy wopółnawowany przez Unię Europiają u mierzyczne i powiecze opowiecznego	

Model V	
Model V	_
Decyzja o budowie oprogramowania Definicja wymagań użytkownika Definicja wymagań restowanie acceptacji użytkowników Definicja wymagań restowanie acceptacji użytkowników Testowanie acceptacji użytkowników Testowanie architektury Testowanie integracji Szczegółowe projektowanie Związek faz projektu z fazami testowania Kodowanie	
Projekt or POLLO. 401 of 2019 (Astations day appoint in entry)s Projekt or POLLO. 401 of 2019 (Astations day appoint in partin a siedary supplifications by praz Unit Europolita et annal foodine Europolitaejo Pundusa Sprincarago APITAL LUZO: APITAL LUZO:	
SANDONII SPANGO FRANCO	26

Aksjomaty testowania		
☐ Testowanie w żaden sposób nie jest w stanie wykazać, że oprogramowanie nie zawiera błędów, a także nie jest w stanie udowodnić, że program będzie zawsze prawidłowo realizował swoje funkcje		
"Testing shows the presence, not the absence of bugs"		
Edsger W. Dijkstra		
Colonieri – nijeljosas imetycijo Projekt nr POKL. 0.10 17 0.0 54710 Kastinioni da popodavli oparieji na wiedzy wogólinaraowany przez Unije Europejską w raznach środnier Europejskogo Funduszu Społecznego szapiska i Pokacia Społecznego szapiska i Pokac	UNIA EMONESIA FINDISCI POLICINI NE CA DETOLATANE 27	

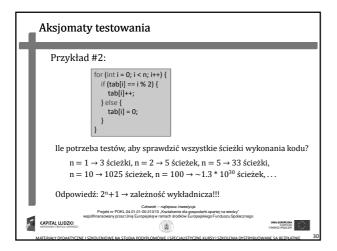






Aksjomaty testowania
☐ Nawet najprostszego programu nie da się przetestować całkowicie (wyczerpująco)
Przyczyny:
 ogromna liczba danych wejściowych i wyjściowych ogromna liczba możliwych ścieżek wykonania kodu programu
 specyfikacja wymagań jest często subiektywna i ocena, czy program działa prawidłowo czy nie, zależy od tego, kto ten program ocenia (użytkownik, obserwator, tester)
definicje miar jakości są subiektywne i często trudno mierzalne
Coloniel – najlepaza inventrycja
Projekt nr POKL D4.01.01-00-213/10 , Kstralkonie dla gespodanki opantej na wiedzy' wspódłinansowany przez Unię Europejską w ramach środków Europejskiego Funduszu Społecznego KAPITAL UDZKI www.sumusowa.
MATERIANY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA BEZPŁATNIE 28

Aksjomaty testowania		
Przykład #1:		
double sum(int a, int b)		
Zakładając, że typ int jest 32-bitowy, ile mamy kombinacji danych wejściowych?		
$(0,0),(0,1),(0,2),\ldots,(0,2^{32}-1),$		
$(1,0),(1,1),(1,2),\ldots,(1,2^{32}-1),$		
$(2,0), (2,1), (2,2), \ldots, (2,2^{32}-1),$		
$(2^{32}-1,0), (2^{32}-1,1), (2^{32}-1,2), \ldots, (2^{32}-1,2^{32}-1)$		
Odpowiedź: $2^{64} = \sim 1.8 * 10^{19}$ (zakładając testowanie z szybkością		
1 test/s, to testowanie zajęłoby ok. 500 miliardów lat)		
Coloxiels – najlepsza investycja Priojekt nr POKL 0.4 0.1 0.1 0.4 0.31310 (Astalbarenia dla gespodarki opantej na wiedzy" węddinansowany przez Unię Europejskiej warzandu frodrike Europejskiejskop Funduszu Społecznego		
KAPITAL LUDZKI ANDORNI TRACIGO PRINCISO FINCATO PRINCIPO FINCAT		

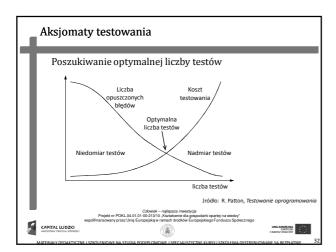








Aksjomaty testowania	
☐ Testowanie oprogramowania jest ryzykowne	
Skoro nie da się przetestować wszystkiego, to trzeba pójść na kompromis i wybrać nieduży, lecz reprezentatywny podzbiór danych do testowania	
Podejmujemy jednocześnie ryzyko, że nie będziemy w stanie wykryć takiego błędu, który objawi się wyłącznie dla takiej kombinacji danych wejściowych, której nie testujemy	
Ile testów potrzeba, aby wykryć jak największą liczbę błędów, zachowując jednocześnie koszty testowania na akceptowalnym poziomie?	
Projekt nr POXLO AL 01-01-00-21910 (Architome dia pappodari oparing na wiedzy' Projekt nr POXLO AL 01-01-00-21910 (Architome dia pappodari oparing na wiedzy' wepółnicznowany przez Unię Europejąka w manach źrodkow Europejakop Fundustu Społecznego www.nachodowany. APPIAR LUDZXI MATERIAN POYNACTYZNE I SYKOL ENIOWE NA STIDIJA POYNYO (MOTOW E I SYKOL INISTYLYNE E I INSYL ISYKOL ENIA (PYSTRAIDINA MATERIAN). 31	



	Aksjomaty testowania
	☐ Im więcej błędów zostało znalezionych, tym więcej błędów pozostaje jeszcze nieznalezionych
	Błędy programistyczne wykazują tendencję do występowania w grupach, co oznacza, że szczególną uwagę należy poświęcać tym fragmentom kodu, w których już wcześniej wykryto jakieś błędy
	Przyczyny:
	 błędy popełniane przez programistę w trakcie pracy zależą od jego sprawności (kondycji) psychicznej, samopoczucia, itp.
	 programiści często powtarzają te same błędy
	 niektóre fragmenty kodu są bardziej podatne na błędy niż inne
	Cołowiek — najlepsza inwestycja Projekt nr POKL 0.4 01.0 10 00 213 10 . Ksztalonenie dla psepodarki opartej na wiedzy' współinansowany przez Unię Europejską w ramań środów Europejskogo Funduszu Społecznego
*	KAPITAL LUDZKI WANDODIN TEREOR SPANSO FARSES PROCESS FARSES
	MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA BEZPŁATNIE 33





Aksjomaty testowania	
Zależność prawdopodobieństwa istnienia błędów jeszcze niewykrytych od liczby błędów dotychczas znalezionych	
yrkych •	
Prawdopodobieństwo istnienia błędów jeszcze niewykrytych	
Prawdopod blędów jes	
liczba dotychczas znalezionych blędów źródio: G. J. Mayers et. al, <i>Sztuka testowania oprogramowania</i> Człowiek – najlepsza inwestycja	
Projek or POKL 64 01 61 00 21310 Kizatkomie dla gopodanki opantija na wiedzy' septificanzowany przez Linje Europojskeja na motorioke Europojskego Funduszu Społecznego KAPITAL LUDZKI KAPITAL LUDZKI MATERIANY DYDAKTYCTNE I STKOLENIOWE NA STUDIA PODPEJOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA BEZPIJATNIE 34.	
Aksjomaty testowania	
☐ Nie wszystkie znalezione błędy zostaną naprawione	
Wykrycie błędu nie oznacza, że automatycznie zostanie on poprawiony	
Przyczyny: • brak czasu (są inne ważniejsze zadania do wykonania)	
 menadżerowie zarządzający projektem mogą podjąć inną decyzję (napięty harmonogram, zbyt duże koszty naprawy) zbyt duże ryzyko związane z koniecznością wprowadzenia bardzo istotnych zmian w oprogramowaniu 	
 nieopłacalność (błąd ujawnia się bardzo rzadko, ma niską "szkodliwość", można go w prosty sposób ominąć) 	-
Projekt or PORL 04 01-0 40-02 130 Novie- nalignosa investigo) Projekt or PORL 04 01-0 40-02 130 Novie- nalignosa dia propodanti oparteri na viestry' wapdifinancowany przez Unię Europajską w ramach środów Europajskągo Funduszu Społecznego «APITAL LUDZKI www.nanosa.eo.eo.eo.eo.eo.eo.eo.eo.eo.eo.eo.eo.eo.	
Aksjomaty testowania	
☐ Trudno jest zawsze jednoznacznie określić, co jest, a co nie jest błędem	
Czy błąd, którego skutków nie można zaobserwować pozostaje błędem?	
Czy zawsze obserwowane zachowanie programu bedzie	

Człowiek – najlepsza inwestycja
Projekt nr POKL.04.01.01-00-213/10 "Kształcenie dla gospodarki opartej na wiedzy"
współfinansowany przez Unię Europejską w ramach środków Europejskiego Funduszu Społecznego

przez wszystkich obserwatorów (programistów, testerów, użytkowników) zinterpretowane w taki sam sposób (jako działanie prawidłowe albo błędne)?







Warunki wykrycia błędu		
Przykład:		
<pre>if (x >= 0) { y = x * x; // blqd - powinno być: y = x + x; } else { y = 0; } return y >= 8;</pre>		
x < 0 → brak osiągalności błędu		
x = 0 lub x = 2 → brak uczulenia błędu		
$x \geq 4 \rightarrow brak$ propagacji (obserwowalności) efektu błędu		
Cobomiés - najdropaza investyroja Projek nr POKL-0.6.1.0 10-0.2110 // katalenia dia gapadanik oparieri na wiedzy' wpojifinansowany przez Unię Europejską w ramach fordiów Europejskago Funduszu Społecznego (APITAL UUDZKI uszciowa raktup plankto **Pasca raktup		

Dokumentowanie błędnego wykonania		
doprowadziły do zad określenie, czy poda podanie alternatywy obserwowalny skute	nrzez: ptkowego wraz z sek poserwowania błędi na sekwencji krokó nych sekwencji kroke podobnych sekwence programu którym uruchamian którym uruchamian któw błędu (np. uszl	cwencją kroków, które nego wykonania programu w jest minimalna ów, dających taki sam iji kroków, które nie powodują y był program kodzonych danych)
Projekt nr POKL.04.01. współfinansowany przez Unię E	Człowiek – najłepsza inwestycja 01-00-213/10 "Kształcenie dla gospodarki uropejską w ramach środków Europejskie	opartej na wiedzy" go Funduszu Społecznego
KAPITAL LUDZKI WASODINA STRATOL SPINISCO MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA.	PODYPLOMOWE I SPECIALISTYCZNE KLI	UNA ELEGELISIA FRANCESCO FRANCESCOLICIO BSY I SZKOLENIA DYSTRYBLIOWANE SA BEZPEATNIE 39







Metodyka testowania oprogramowania	
□ Elementy systematycznej metodyki procesu testowania: • określenie przesłanek i wymagań dotyczących generowanych testów • wybór strategii testowania • projektowanie testów (opracowanie i weryfikacja specyfikacji testów) • planowanie procesu testowania • wzbogacanie testów poprzez inspekcję kodu • wybranie (ewentualnie opracowanie) narzędzi wspomagających testowanie • implementacja testów • dokumentowanie testowania • ocena testów, identyfikowanie pominiętych przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów • Ochowanie Populacji przesłanek testowania w celu udoskonalenia testów	
	1
Metodyka testowania oprogramowania Testowanie możemy podzielić ze względu na: sposób przeprowadzania testowania zakres testowania przeznaczenie testowania	
 □ Podział testów ze względu na sposób ich przeprowadzenia: • manualnie - przez testera wykonującego zaplanowaną sekwencję operacji • automatycznie - za pomocą odpowiednich narzędzi (testing framework), pozwalających na uruchomienie napisanych testów lub odtworzenie testów uprzednio nagranych 	
Projek in POKL 64.01.01-00-21310 /Kauthonie dia geocolari quartej na viedzy' supdifinazionamy przez Usię Europejaką w reducción Europejakop Fundaces Spekecznego CAPITAL WOZE MATERIAŁY PROMOTYCZNE I SZKOLENIOWE NA STUDIA PODYPI OMONE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBLIOWANE SA BEPPARTNE 41	
Metodyka testowania oprogramowania	
 Podział testów ze względu na zakres testowania: testy specyfikacji (specification testing) – pozwalają sprawdzić statycznie poprawność opracowanych lub dostarczonych specyfikacji (wymagań, danych, projektu) 	
 testy modułów, testy jednostkowe (unit testing) – testują one system na poziomie pojedynczych funkcji (metod) testy integracyjne (integration testing) – pozwalają sprawdzić poprawność współpracy różnych komponentów składowych systemu 	
 testy systemowe (system testing) – dotyczą testowania aplikacji jako całości, zwłaszcza w kontekście weryfikacji specyfikacji wymagań testy akceptacyjne (acceptance testing) – służą do oceny zgodności działania programu z wymaganiami użytkownika 	
Catowiet - najkapaza investrycja Projekt nr POKL 04.01.0-00.219.10 /kstalacene dla gospodani opartej na wiedzy vespóllmarsowany przez Unię Europejską w namach szotkow Europejskop Funduszu Społecznego KAPITAL IUDZKI	



KAPITAL LUDZKI





Metodyka testowania oprogramowania	
 Podział testów ze względu na ich przeznaczenie: testy funkcjonalne (functional testing) - pozwalają ocenić, czy oprogramowanie działa zgodnie ze specyfikacją 	
 testy regresyjne (regression testing) – służą do sprawdzenia, czy wprowadzone w kodzie modyfikacje bądź zmiany w środowisku, nie skutkują powstaniem nowych błędów lub ujawnieniem się błędów dotychczas niewykrytych 	
 testy przenaszalności/testy konfiguracji (portability/configuration testing) – pozwalają ocenić, czy oprogramowanie będzie działać na różnych platformach sprzętowych lub systemach operacyjnych 	
 testy post-awaryjne (disaster recovery testing) – służą ocenie zachowania aplikacji po wystąpieniu sytuacji awaryjnej testy bezpieczeństwa (safety testing) – służą ocenie bezpieczeństwa aplikacji 	
Projekt in POCL 0.0 (1) 10 (2) 110 (Asterbase Interestry) Projekt in POCL 0.0 (1) 10 (2) 110 (Asterbase Interestry) Septimization of the Europeaka in Septimization of the Eur	
Metodyka testowania oprogramowania	
☐ Przed rozpoczęciem testowania konieczne jest przyjęcie pewnych założeń dotyczących wyboru strategii (koncepcji) testowania programu i projektowania przypadków testowych	
Najczęściej stosowane strategie testowania: testowanie funkcjonalne, zwane także testowaniem metodą "czarnej skrzynki" (black-box testing), testowaniem sterowanym danymi	
(data-driven testing) lub testowaniem wejścia-wyjścia (input/output testing) • testowanie strukturalne , zwane także testowaniem metodą "białej	
skrzynki" (<i>white-box testing</i>), metodą "szklanej skrzynki" (<i>glass-box testing</i>) lub testowaniem sterowanym logiką (<i>logic-driven testing</i>) • testowanie oparte na doświadczeniu (<i>experience-based testing</i>)	
Projekt in POKL 04.0 (1) 10 05 21 10 / Austheria de gampodari opante in a viedzy septimizarionary przez Unię Europejską w ramach sociolow Europejskop Funduszu Spiecznego CAPITAL LUDZXI «MACHON MACIA POKLA" STYLO ENINDE MA STUDIA SOTYNO GAMPAE I SOSIFIANI STYLO FANI I STYLO ENINDE MA STUDIA SOTYNO GAMPAE I SOSIFIANI STYLO ENINDE STANDARD STANDA	
Testowanie funkcjonalne (<i>black-box</i>)	
☐ Podstawowym założeniem testowania funkcjonalnego jest znajomość wymagań dla testowanej funkcjonalności	
☐ System traktowany jest jak "czarna skrzynka", o której nic nie wiemy, a testowanie polega na wprowadzeniu na jej wejście danych testowych i porównaniu otrzymanych wyników	
z danymi oczekiwanymi Testowany system	
Dane wejściowe (testowe) P Dane wyjściowe	
Catowiek – najlepaza inwestycja Projekt nr POKL 0.4 01.01 -00 213/10 . Kształcenie dla opcodarki opartei na wiedzn*	

Człowiek – najlepsza inwestycja



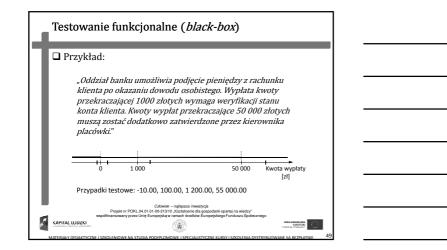


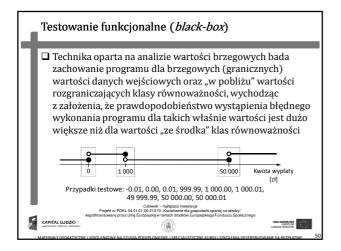


Testowanie funkcjonalne (<i>black-box</i>)	
☐ Zadanie: Zaprojektuj przypadki testowe, umożliwiające przetestowanie następującego programu:	
"Program odczytuje trzy liczby całkowite, które stanowić mają długości boków trójkąta. Celem programu jest wypisanie komunikatu informującego, czy mamy do czynienia z trójkątem	
równobocznym, równoramiennym (dwa równe boki i trzeci różny od nich) czy też różnobocznym (czyli takim, którego wszystkie boki mają różne długości)."*	
*źródło: G. J. Mayers, et al., <i>Sztuka testowania oprogramowania</i>	
Project in POXL D4 (1) in 05 of 25 11/0 (Asserbed de proposité opartie in a viectry supdifinationary przez Unije Europejska w amach excitow Europejskago Funduszu Społecznego wpolifinationary przez Unije Europejska w amach excitow Europejskago Funduszu Społecznego (SP) CAPITAL LUDZXI MATERIAN DYNANTYCNE I SZNOLENIOWE NA STUDIA PODVPO/MOWNE I SESCUAISTYCZNE KURSY I SZNOLENIA DYSTRYBUOVANIE SA BEZPATNIE 46	
	1
Testowanie funkcjonalne (<i>black-box</i>)	
☐ Techniki testowania funkcjonalnego wykorzystują: • podział na klasy równoważności (equivalence partitioning) • analize wartości brzegowych (boundary value analysis)	
 tablice przejść stanów (state transition testing) tablice decyzyjne (decision table testing) lub grafy przyczynowoskutkowe (cause-effect graphs) 	
 przypadki użycia (use case testing) kontrolę składni (syntax testing) 	
 kombinacje par parametrów wejściowych (pairwise testing) losowy lub pseudo-losowy dobór danych testowych (random testing) 	
Projek nr POKL 04.01.01-00-21310 Kataleone dia geopodarki opartiji na wiedzy' wopółnianowany przez Utie Izenopicjak w manach sokróle Europejskop Funduszu Społocznego	
CAPITAL LUDZKI SANCOSA SHARISA SPANGO MATERIANY DYDAKTYCZNE I STYCH ENIOWE NA STLIDIA PODVRI OMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBLIOWANE SA BEZPIATNE 47	
_ Testowanie funkcjonalne (<i>black-box</i>)	
restowante funkcjonante (black box)	
☐ Technika oparta na podziale na klasy równoważności zakłada podział dziedziny wartości danych wejściowych na skończona liczbe podzbiorów (klas równoważności)	
w taki sposób, aby: program zachowywał się analogicznie dla wszystkich wartości danych	
z tej samej klasy zastąpienie testowania wszystkich wartości danych z klasy równoważności przez testowanie pojedynczej (reprezentatywnej) wartości danej z tej klasy, pozwalało na wykrycie błędu z takim	
samym prawdopodobieństwem Przypadki testowe projektuje się tak, aby pokrywały wszystkie możliwe scenariusze	
Culowiel – najlepsza inwestycja Projekt nr POKL 04.01.01.00.2130 (Kastilevenia dla gespotianki opartiej na wiestry) wepolifinansowany przez Unię Europejskiej w namach środkie Europejskiego Funduszu Społecznego KAPITAL UDZKI AMORIONAL OD KARTINIA SPORECKO (KARTINIA SPORECKO) West Namach (Karti	









Testowanie funkcjonalne (<i>black-box</i>)							
☐ Technika wykorzystująca tablice przejść między stanami służy do testowania systemów wyspecyfikowanych lub modelowanych jako maszyna stanowa							
Przypadki testowe projektuje się tak, aby sprawdzały osiągalność poszczególnych stanów, a także dozwolone oraz niedozwolone przejścia między stanami							
Przykład:							
0 S ₁ S ₂ S ₂ S ₂ S ₁							
Całowiek – najłopaza inwestycja Projekt nr POKL 04.01.01-02-13/10 (Kaztalcenia dla gospodarki opartej na wiedzy' współfinansowany przez Unię Europejską w manach śrotókow Europejskego Funduszu Spółecznego							
APPITAL LUDZKI MATERIAN PRODUCTVZNE I SZKOL ENIOWE NA STLINIA PODVED OMNOWE I SOFCIALISTYCZNE KLIBSY I SZKOL ENIA DYSTRYBLIOWANE SA REPARTNE 51							







Testowanie funkcjonalne (<i>black-box</i>)	
☐ Technika wykorzystująca tablice decyzyjne lub grafy przyczynowo-skutkowe bada zachowanie programu dla różnych kombinacji logicznych warunków wejściowych (decyzji logicznych)	
Podejście to wymaga ujęcia specyfikacji działania programu jako logicznej sieci kombinatorycznej, przedstawionej albo bezpośrednio w postaci tablicy decyzyjnej, albo w postaci grafu przyczynowo-skutkowego, który następnie przekształca się w równoważną mu tablicę decyzyjną, wykorzystując tzw. technike śledzenia wstecznego	
Projekt nr POKL OM 01-0-00-21310 (Astaleone dia geogodanii opartinji na wiedzy współlinarowany przez Unię Europojskia pra nacion i scolonii Europojskia pri unicazui Społecznego (APITAL LUDZXI) MATERIAN OROANTYCZNE I SZYOLINIOME NA STUDIA POORPO-OMONE I SZYCLANIKTYCZNE KURKY I SZYOLINIA OYSTREBIOWAME SA 86 ZPRATNIE 52.	

Testowanie funkcjonalne (<i>black-box</i>)										
Przykład:										
Graf przyczynowo-skutkowy			Tab	lica (decy	zyjna	a			
	C_1	F	F	F	F	Т	Т	Т	Т	
	C ₂	F	F	Т	Т	F	F	Т	Т	ĺ
V M_1 D_1	C ₃	F	Т	F	Т	F	Т	F	Т	ĺ
(c_2)	M_1	Т	Т	F	F	Т	Т	Т	Т	1
^ (D ₂)	D_1	F	F	Т	Т	F	F	F	F	1
C_3	D_2	F	Т	F	F	F	Т	F	Т	ĺ
Przypadki testowe dla decyzji D ₂ = False (F)										
Projekt nr POVL.04.0. Caloniek – najlegazar inwestycja: Projekt nr POVL.04.0.101-0-0-0-131-10 (Actalidennie dia propodenki opanlej na wiedzy' recodificial projekt nr przez Uso.01.01-0-0-0-131-10 (Actalidennie dia propodenki opanlej na wiedzy' recodificial projekt nr przez Uso.01.01-0										
KAPITAL LUDZKI KAPDOMI STRATICII SPRINGICI	(m)						0.40	EUROPEJSI EUROPEJS SZ POLICZ		9

Testowanie funkcjonalne (<i>black-box</i>)
□ Technika wykorzystująca przypadki użycia lub scenariusze zastosowania pozwala na tworzenie przypadków testowych na podstawie informacji o programie z punktu widzenia użytkownika Przypadki użycia opisują interakcje między aktorami a systemem (sekwencje działań użytkowników mające określony cel biznesowy) i są użyteczne do sprawdzania poprawności działania programu, związanego z typowymi aktywnościami użytkowników (badanie przepływów, warunków początkowych i końcowych)
Project or POXLO. 0.10 oct 0.0190 (Actioned - applicate investor): Project or POXLO. 0.10 oct 0.01910 (Actioned appropriate deproparative opartiq in a windry woodfinancewary prezz Unit Europejskiq or manch forcifice Europejskiq or Junior State (Application Action Europejskiq or State (Application Action Europejskiq or State (Application Europejskiq or State (Application Europejskiq or State (Application Europejskiq or State (Application Europejskiq or







Testowanie funkcjonalne (blac	k-box)		
☐ Technika wykorzystująca kontrol projektowania przypadków testo dziedziny danych wejściowych i/	wych na podstawie definicji		
wyjściowych			
Kontrolę składni stosuje się najcz poprawności danych złożonych (ł plików, struktury rekordów), dla	łańcuchy znaków, formaty których można zdefiniować		
formalne reguły sprawdzania ich gramatyk formalnych, wyrażeń re			
Przykład: ^[+-]?[a-zA-Z]+\.\d{2}-\d{3}\$			
CAPITAL LUZXI KAPITAL LUZXI CAPITAL	popularii oparteij na wiedzy" opojiskiego Funduszu Społecznego www.suswestawa		
Testowanie funkcjonalne (<i>blac</i>	rk-box)		
☐ Technika wykorzystująca kombin wejściowych zakłada taki sposób	acje par parametrów projektowania przypadków		
☐ Technika wykorzystująca kombin	nacje par parametrów projektowania przypadków zystkie możliwe, dyskretne		
☐ Technika wykorzystująca kombin wejściowych zakłada taki sposób testowych, aby przetestować wsz kombinacje każdej pary parametr Najprostsze błędy w programie sz z przetwarzaniem pojedynczego najczęściej występującą przyczyn	nacje par parametrów projektowania przypadków zystkie możliwe, dyskretne rów wejściowych ą najczęściej związane parametru; drugą ną najprostszych błędów		
☐ Technika wykorzystująca kombin wejściowych zakłada taki sposób testowych, aby przetestować wsz kombinacje każdej pary parameti Najprostsze błędy w programie sz z przetwarzaniem pojedynczego	nacje par parametrów projektowania przypadków zystkie możliwe, dyskretne rów wejściowych ą najczęściej związane parametru; drugą ią najprostszych błędów iiędzy parami parametrów		
☐ Technika wykorzystująca kombin wejściowych zakłada taki sposób testowych, aby przetestować wsz kombinacje każdej pary parametr. Najprostsze błędy w programie sz przetwarzaniem pojedynczego najczęściej występującą przyczyn są nieprawidłowe interakcje pom	nacje par parametrów projektowania przypadków zystkie możliwe, dyskretne rów wejściowych ą najczęściej związane parametru; drugą ną najprostszych błędów niędzy parami parametrów rć stosując tę technikę dla testowania losowego		
☐ Technika wykorzystująca kombin wejściowych zakłada taki sposób testowych, aby przetestować wsz kombinacje każdej pary parametr Najprostsze błędy w programie sz przetwarzaniem pojedynczego najczęściej występującą przyczyn są nieprawidłowe interakcje pom wejściowych, które można wykry Technika ta stanowi alternatywę	nacje par parametrów projektowania przypadków zystkie możliwe, dyskretne rów wejściowych ą najczęściej związane parametru; drugą ią najprostszych błędów ijedzy parami parametrów rć stosując tę technikę dla testowania losowego dla niewielkich problemów		

Testov	Testowanie funkcjonalne (<i>black-box</i>)								
Przy	Przykład:								
Pro	gran	o ti	rzeci	h binarnych parameti	rach	wej	ścio	wych: A, B i C	
	Α	В	С	← Wszystkie kombina	cje d	anyc	h we	jściowych: 2 ³ = 8	
	0	0	0						
	0	0	1	Pr	zypa	dki te	estov	ve:	
	0	1	0		Α	В	С		
	0	1	1		0	0	1		
	1	0	0		0	1	0		
	1	0	1		1	0	0		
	1	1	0		1	1	1		
	1	1	1						
MARODONIA STRATEGIA SP	Codosido - nagigazaz investroja Projekt nr POKL.04.01.01-00-01310 (Actalement dia propodanti opantej na wiedzy woopdiffinancementy presz Liniq Europejda w remain fordilne Europejdango Funduccio Spridocznego Wana Bendrian Wana								







	_
Testowanie funkcjonalne (<i>black-box</i>)	
☐ Technika oparta na losowym (pseudo-losowym) doborze danych testowych wykorzystuje pseudolosowy algorytm generujący ciąg danych na podstawie pewnej uprzednio przygotowanej sekwencji lub rozkładu wartości	
Podejście to jest wykorzystywane do testowania wymagań niefunkcjonalnych takich, jak wydajność lub niezawodność	
Odmianą tego typu podejścia w odniesieniu do interfejsu użytkownika jest tzw. "małpie testowanie" – metoda testowania programu polegająca na losowym naciskaniu klawiszy oraz losowym poruszaniu i klikaniu myszką	
Projek nr POKL 0.4 0.1 00-02-10 najlipozaz inventycja Projek nr POKL 0.4 0.1 0.0 02-10 n.0 02-20 n.0 n.0 n.0 03-20 n.0 n.0 n.0 03-20 n.0 n.0 n.0 03-20 n.0 n.0 n.0 03-20 n.0 n.0 n.0 n.0 n.0 n.0 n.0 n.0 n.0 n.	
MATERIALY DIDAKTYCZNE I SZKICIANOWE NA STUDIA PODIPICAMOWE I SPECIALISTYCZNE KURSY I SZKICIANIA DYSTRYBLOWANI SA BIZPIATNIE 58.	
Testowanie funkcjonalne (<i>black-box</i>)	
☐ Podsumowanie: ■ testy funkcjonalne są powtarzalne ■ testy funkcjonalne nadają się do wielokrotnego użycia ■ testowanie funkcjonalne można stosunkowo łatwo zautomatyzować ■ testowany jest nie tylko sam program, lecz jednocześnie środowisko,	
w którym się wykonuje nie wszystkie aspekty działania programu można w ten sposób przetestować wyniki testów mogą nie być do końca wiarygodne (niedoszacowanie liczby błędów)	
 przyczyny i lokalizacje odnalezionych błędów nie są znane 	
Coloniale - realisposa investigation investigation investigation investigation investigation investigation in a visidary Projekt nr POKIL OAL 0.10 d.0 10 d.0 024 110 / Kastellande and grapposate in quantum investigation in a visidary responsation investigation investigation investigation in a visidary responsation investigation inves	
Testowanie strukturalne (<i>white-box</i>)	
☐ Podstawowym założeniem testowania strukturalnego jest znajomość sposobu implementacji testowanych funkcji	
☐ Testy opracowywane są na podstawie wiedzy o kodzie źródłowym (strukturze programu), a testowanie polega na wprowadzeniu na wejście danych testowych i porównaniu	
otrzymanych wyników z danymi oczekiwanymi Testowana funkcja	
Dane wejściowe Dane wejściowe wyjściowe	
(testowe) }	







Testowanie strukturalne (<i>white-box</i>)	
□ Podstawowym ograniczeniem testów strukturalnych jest to, że nie są one w stanie wykryć błędów wynikających z braku funkcjonalności programu □ Testowanie strukturalne można podzielić na: • testowanie strukturalne można podzielić na: • testowanie dynamiczne (wymagające uruchomienia programu) • testowanie statyczne (nie wymagające uruchomienia programu) □ Ogólna strategia przeprowadzania większości testów strukturalnych polega na testowaniu istniejącego kodu, aby: • każda instrukcja zawarta w kodzie źródłowym została wykonana co najmniej jeden raz • niezależnie przetestować każdą funkcję (metodę, procedurę) zdefiniowaną w kodzie źródłowym Colombia - najparan immetrycja APPIIAK LUDZXI MARTERMAY DYDMATYCZNE I SZROLENIA DYSTRYBLIOWANE SA BE ZPŁATNIE 61	
	1
Testowanie strukturalne (<i>white-box</i>)	
□ Techniki testowania strukturalnego opierają się na analizie: • przepływu danych • pokrycia kodu • strukturalnej kodu □ Analiza przepływu danych polega na sprawdzeniu sposobu przetwarzania danych wejściowych przez poszczególne fragmenty kodu programu (funkcje, moduły) Porównanie wartości wybranych zmiennych programu na różnych etapach przetwarzania, z wartościami oczekiwanymi lub wartościami niedozwolonymi umożliwia wykrycie i przybliżone zlokalizowanie błędu □ Projekt r POKL OL 01.01 (20 20 13 10 (Kathadonia) (20 20 20 13 10 (Kathadonia) (20 20 20 13 10 (Kathadonia) (20 20 20 20 20 20 20 20 20 20 20 20 20 2	
Testowanie strukturalne (<i>white-box</i>)	
□ Analiza pokrycia kodu służy do projektowania przypadków testowych, przeznaczonych do testowania wewnętrznej struktury programu w taki sposób, aby były one w stanie wykonać jak największą część testowanego kodu □ Analiza pokrycia kodu może być przeprowadzana z użyciem: • programów śledzących (debugerów), pozwalających obserwować, które instrukcje kodu są wykonywane w trakcie przetwarzania danych testowych • analizatorów pokrycia kodu, które automatyzują proces zbierania informacji o pokryciu kodu przez poszczególne testy, wyświetlają szczegółowe statystyki dotyczące testowania oraz pozwalają na uzyskanie informacji, która część kodu nie została pokryta przez zastosowane testy	







7	estowanie strukturalne (<i>white-box</i>)
Ţ	3 Stopień pokrycia kodu jest miarą oceny jakości zestawu
	testów, pozwalającą na:
ı	 oszacowanie, czy program został przetestowany w wystarczającym stopniu
	 znalezienie minimalnego podzbioru testów (np. do testowania regresyjnego)
	Podstawowe kryteria projektowania testów pokrycia kodu:
	 pokrycie wyrażeń (statement coverage)
	 pokrycie rozgałęzień (branch coverage)
	 pokrycie ścieżek (path coverage)
	 pokrycie decyzji (decision coverage)
	 pokrycie warunków logicznych (branch condition coverage)
_	Człowiek – najłępsza inwestycja Projekt nr POKL 04.01-0-0-213/10 "Ksztalcenie dla gospodarki opartej na wiedzy" współfinansowany przez Unie Europejska w ramach środków Europejskiago Funduszu Społecznego
S.	APITAL LUDZKI ADMINISTRAÇÃO PRINCIPLO PRINCIP
MA	ERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA REZPŁATNIE 64

_	Testowa	nie strukturalne (<i>white-box</i>)	
	doborz	krycia wyrażeń (instrukcji programu) po e danych testowych, aby każda instrukcja sterowania została wykonana co najmnie	wynikająca
	instruk instruk uzyska instruk	pu testowanie nie bierze pod uwagę zależ cji sterujących (pętle, instrukcje warunko cje wyboru) od przetwarzanych danych i nie w nim 100% pokrycia (tj. wykonanie cji programu) nie oznacza przetestowani ych ścieżek wykonania programu	we, dlatego każdej
	źródłov	ażda instrukcja znajduje się w osobnej lir wego, to taki test jest równoważny testow y programu (linii kodu)	
		Projekt nr POKL-04.01.01-00-213/10 "Ksztalcenie dla gospodarki opartej na wiedzy" współfinansowany przez Unię Europejską w ramach środków Europejskiego Funduszu Społecznego	
1	KAPITAŁ LUDZKI KARODOMA STRATĘGIŁ SPENOŚCI	ů.	UNIA EUROPESIA EUROPESI EUROPESI POLICZNY
l	A A ATTERNATIVE DIVER A MEDICENIA	E I STACH ENIONNE NA CELIDIA DODADINE I COECIALISTACANE MIDEA I CAMOLENIA DASTRADA	

Testowanie strukturalne (<i>white-box</i>)							
Przy	kład:						
	1: a = 10; 2: if (x > 0 && y < 5) 3: y = a * x; 4: else 5: y = a - x; 6: a += x;	No	Yes y = a * x				
	Jak dobrać dane testowe (x, y), aby uzyskać jak największe pokrycie instrukcji?						
Test TC1: $\{x = 1, y = 2\}$ pokrywa instrukcje w liniach: 1, 2, 3 i 6 Test TC2: $\{x = 0, y = 0\}$ pokrywa instrukcje w liniach: 1, 2, 4, 5 i 6							
Odp	Odpowiedź: Testy TC1 i TC2 pozwalają na uzyskanie 100% pokrycia instrukcji						
KAPITAŁ LUDZK GAŁOCINA STATECI SPO	Projekt nr POKL.04.01.01-00-213/10 "Ks współfinansowany przez Unię Europejską w ram	elepsza inwestycja ztalecenie dla gospodarki opartej na wiedzy" ach środków Europejskiego Funduszu Społecznego	UNIA EUROPEJSIA EUROPEJSI FAROISZ SPOLECKY				







	Testowanie strukturalne (<i>white-box</i>)		
	☐ Test pokrycia rozgałęzień polega na takim doborze danych testowych, aby wykonać co najmniej jeden raz każdą krawędź w grafie przepływu sterowania (tzn. zapewnić przejście przez wszystkie możliwe rozgałęzienia wynikające z instrukcji warunkowych, pętli oraz instrukcji wyboru)		
	Każde rozgałęzienie w grafie sterowania traktowane jest w tym teście niezależnie od pozostałych		
	Wynikiem instrukcji warunkowej i instrukcji pętli są dwa rozgałęzienia, zaś instrukcji wyboru tyle rozgałęzień, ile jest możliwych opcji wyboru		
	Test pokrycia rozgałęzień stanowi uogólnienie testu pokrycia wyrażeń Czisak - najpaca inwebyci Projek ir POLL 64 D 10 00 219 0 Ketalome da genedowi opatej na wiedzy wpolforacomowy przez Unie Europsyka Ciele Europsyka po Fudeza Spelecnego		
100	CAPITAL LUDZKI WASTERNAY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODVPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANI S AB BZPŁATNIE 67		

Testowanie strukturalne (<i>white-box</i>)				
Т	Przykład:			
	1: a = -1; 2: if (x > 0) 3: a = 5; 4: if (y < 9) 5: a = 0; 6: else 7: a += 10;	ves ves		
	Jakie jest pokrycie rozgałęzień dla testów: TC1: $\{x = 1, y = 0\}$ i TC2: $\{x = 5, y = 10\}$?			
	Odpowiedź: Testy TC1 i TC2 pozwo uzyskać 4/5 = 80% pokrycia rozgałę (100% pokrycia instrukcji)			
	Projekt nr POKL.04.01.01-00-213/10 "Ks	ajlepsza inmestycja zastalacnie dla ospopodarki opartej na włedzy mach środków Europejskiego Funduszu Społecznego uww.eagggwa Facetz Pouczer		

Testowanie strukturalne (<i>white-box</i>)			
☐ Test pokrycia ścieżek polega na t testowych, aby w miarę możliwo ścieżki w grafie przepływu stero początkowego do określonego w	ści wykonać wszystkie wania od węzła		
Ograniczenia badania pokrycia ś • liczba ścieżek rośnie wykładniczo w (rozgałęzień w grafie przepływu sto może być nawet nieskończona	vraz z liczbą decyzji w programie		
 z uwagi na występujące warunki log w której nie wszystkie ścieżki będą uzyskanie 100% pokrycia ścieżek 	, , , , ,		
Projekt in PORLO 61 (1) GODIST - Aglippaza (invest) in PORLO 61 (1	ospodarki opartej na wiedzy' uropejskiego Funduszu Społecznego uwa sacrosoa paces policznego		







Testowanie strukturalne (<i>white-box</i>)	
Uwzględniając powyższe ograniczenie, kompletny test pokrycia ścieżek zastępuje się najczęściej testowaniem odpowiednio dobranego (reprezentatywnego) podzbioru ścieżek	
W przypadku testowania pętli liczbę ścieżek ogranicza się w taki sposób, aby uwzględnić następujące przypadki: żadna iteracja pętli nie zostanie wykonywana	
 zostanie wykonane n iteracji pętli (n - parametr) w taki sposób, aby wszystkie ścieżki wewnątrz pętli zostały co najmniej raz wykonane zostanie wykonana minimalna liczba iteracji, przeciętna liczba iteracji 	
oraz maksymalna liczba iteracji	
Projekt or POKLA 6.0 (0.00 2319 (Astanless) and apoptatel inquirity in windry wopdificationary przet Unit Europejal w remain worden Europejalege Fundates Spidecringo (APITA: LUDZI MATERIAN PRIVATVYZNE I SYKOI ENIOWE NA STIDIJA POTVYY (MOTUNE) ESPECIALISTY/YNE ELIESY I SYKOI ENIA (INSTRUMENTAL SIDIJA) POTVYZNE FURSY I SYKOI ENIA (INSTRUMENTAL SIDIJA) POTVYZNE ELIESY I SYKOI ENIOWE NA STIDIJA POTVYZNE P	

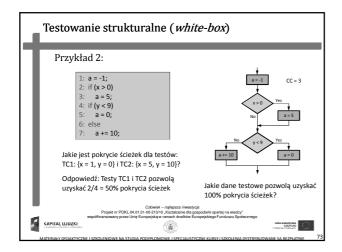
Testowanie strukturalne (<i>white-box</i>)			
Test pokrycia ścieżek wykorzystujący podejście LCSAJ (Linear Code Sequence And Jump) bierze pod uwagę wszystkie ścieżki bazujące na liniowych sekwencjach kodu które są definiowane przez trójki punktów w kodzie źródłowym, wskazujące odpowiednio: początek sekwencji instrukcji, koniec tej sekwencji oraz docelową instrukcję, d której będzie przekazane sterowanie		erze pod uwagę zch sekwencjach kodu, iktów w kodzie o: początek sekwencji locelową instrukcję, do	
Stopień skomplikowania testowanego fragmentu kodu odzwierciedla tzw. złożoność cyklomatyczna CC (cyclomatic complexity), która określa liczbę niezależnych ścieżek w programie jako d+1, gdzie d jest liczbą binarnych wezłów decyzyjnych			
Cclowiek - najlepsza innestycje Projekt nr POKL.04.01.01-00.213/10 "Kstatiscenie olia gospodanki opastej na wiedzy' współfinansowany przez Uhię Europejskieg w ramach środków Europejskiego Funduszu Społecznego			
KAPITAŁ LUDZKI VARODONIA STRATICIA SPONOSCI MATERIAŁY DYDAKTYCZNE I SZKOI ENIOWE NJ.	n	UNIX BURDOUSA COMPANIA COMPANI	

Testowanie strukturalne (white-box)			
Przykład 1			
1: if (su 2: a 3: a++; 4: if (su 5: a	= 1;	Success No are	Yes a = 1
Jakie największe pokrycie ścieżek można uzyskać dla powyższego fragmentu kodu?			
Odpowiedź: Niezależnie od wartości CC=3			
parametru "success" zawsze uzyskamy 2/4 = 50% pokrycia ścieżek			
Człowiek – najlięcaza inwestycja Projek nr POKL DA 10-10-02 31/3 U Katalacenie dla gospodarki opartej na wiedzy* wodfinansowany przez Unie Europejska v ramań brodków Europejskop Funduszu Społocznego			
KAPITAŁ LUDZKI NAKODOBIA STRATECIA SPONOSCI	ű.	i	UNIA EUROPESSA SUROPESSO FUNOSSE POLICENY
-	CAUGUSE NA CEUDIA DODUM DA AGUSE	I COECUALICTYCZNIE WLIDEV I CZWOLENIA DWETDYDLIOI	MANUS CA DETRU ATAUS









Testowanie strukturalne (<i>white-box</i>)				
☐ Test pokrycia decyzji polega na takim doborze danych testowych, aby niezależnie od siebie przetestować warunki logiczne w instrukcjach sterujących				
Bez względu na to, czy warunek logiczny jest prosty czy złożony, sposób jego testowania polega na tym, aby jego wynik raz przyjął wartość logicznej prawdy i raz fałszu Przykład:				
			1: if (x >= 1 && y < 5) 2: a = 0;	
TC1: {x = 0, y = 1}, TC2: {x = 1, y = 1} – 100% pokrycie decyzji				
Coloxiek – najlepaza investycja Projek nr POKLO.4.01.01-00.21310 "Katalicenie dla gospodani opastej na wiedzy' współlinastowany przez Unije Europejskie w march śrotókie Europejskiego Fundusza Społecznego				
CAPITAL LUDZIXI ANDORNI TEXTILE PRINCED TEXTILE PRINCED				

	Testowanie strukturalne (<i>white-box</i>)			
1	☐ Test pokrycia warunków logicznych polega na takim doborze danych testowych, aby przetestować złożone warunki logiczne w instrukcjach warunkowych			
		y testowania złożonych warunków logiczi	,	
	 prosty test pokrycia warunków – testowany jest każdy warunek elementarny tak, aby raz przyjął wartość logicznej prawdy i raz fałszu 			
	 test wielokrotnego pokrycia warunków – testowana jest każda możliwa kombinacja wyników warunków elementarnych (dla n warunków elementarnych mamy 2ⁿ kombinacji) 			
	 minimalny test wielokrotnego pokrycia warunków – testowany jest podzbiór możliwych kombinacji wyników warunków elementarnych wynikający z zasady warunkowego obliczania wartości wyrażenia 			
	(sho.	rt circuit evaluation) Człowiek – najlepsza inwestycja		
e.	KAPITAŁ LUDZKI MARODOMA STRATEGIA SPENOSCI	Projekt nr POKL 04.01.01-00-213/10. Kaztalconie dla gespodarki opantej na wiedzy' współfinansowany przez Unię Europejską w ramach środków Europejskiego Funduszu Społecznego (II)	UNIA EUROPLISAA BUROPLIA FJADESE POLICZIY	



KAPITAL LUDZKI





Testowanie strukturalne (<i>white-box</i>)	
Przykład:	
1: if (x > 0 y < 9) 2: a = 5;	
Zaprojektuj dane testowe (x, y), aby uzyskać jak największe pokrycie w:	
a) prostym teście pokrycia warunków b) teście wielokrotnego pokrycia warunków c) minimalnym teście wielokrotnego pokrycia warunków	
Odpowiedź:	_
a) TC1: (x = 0, y = 10), TC2: (x = 1, y = 0) – pokrycie 100% b) TC1: (x = 0, y = 10), TC2: (x = 0, y = 1), TC3: (x = 2, y = 10), TC4: (x = 2, y = 1) – pokrycie 100%	
c) TC1: {x = 0, y = 10}, TC2: {x = 0, y = 1}, TC3: {x = 2, y = 10} – pokrycie 100%	
Projekt nr POKL 04.01.09.02.130 (Assistance del grapopoleni opartini na viestry septimi na viestra septimi na viestry septimi n	
MATERIANY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA RODDYROMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA BEZPJATNIE 7	
_ Testowanie strukturalne (<i>white-box</i>)]
restorante strukturante (water 50%)	
☐ Podsumowanie: • testy strukturalne oceniają tylko istniejący kod programu i należy je	
traktować jako uzupełnienie, a nie zastąpienie testów funkcjonalnych testy strukturalne pozwalają ustalić przyczynę i lokalizację błędów	
 testy strukturalne mogą być wykorzystane do optymalizacji działania programu 	<u> </u>
 testy pokrycia nie oceniają efektywności samych przypadków testowych, lecz służą głównie do oszacowania, jaka część kodu 	
została przez nie sprawdzona (nawet 100% pokrycie w najbardziej zaawansowanych testach nie gwarantuje poprawności kodu)	
 znajomość kodu źródłowego może być bardzo użyteczna podczas konstruowania przypadków testowych (np. do ustalenia wartości 	
granicznych testowanych parametrów) Cdowiek - ngilopza inwentycja	
Projekt nr POKL 04.01.01 04.02 213 01 (Australes and papedani opartie) na wiedzy współlinianzowany przez Unię Europejskiej w ramach dockier Europejskiejo Pundusu Społecznego KAPITAL LUDZXI 44400646 PUNDUS	
MATERIAN DYNATYCZNE I SZNOLENIOWE NA STLIDIA PODDYROMOWE I SPECIALISTYCZNE KURSY I SZNOLENIA DYSTRYBLIOWANE SA BEZPATNIE 7	<u> </u>
	_
Testowanie oparte na doświadczeniu	
☐ Testowanie oparte na doświadczeniu jest odmiana	<u> </u>
testowania strukturalnego, które wykorzystuje wiedzę, intuicję oraz wcześniejsze doświadczenia testera, w celu	
poszukiwania błędów w programie w oparciu o takie kryteria, które nie zawsze można formalnie zdefiniować	
Do podstawowych technik testowania, które wykorzystują doświadczenie testera należa:	
odgadywanie błędów (<i>error guessing</i>) testowanie bazujące na listach kontrolnych (<i>checklist-based</i>)	
 testowanie eksploracyjne (exploratory testing) 	
■ przeprowadzanie ataków (<i>attacks</i>)	

Człowiek – najlepsza inwestycja







Testowanie oparte na doświadczeniu	
□ Zgadywanie błędów jest techniką polegającą na takim projektowaniu przypadków testowych, aby sprawdzić te miejsca w programie, które zdaniem testera są najbardziej narażone na obecność błędów (np. implementacje bardziej skomplikowanych algorytmów) □ Testowanie bazujące na listach kontrolnych polega na weryfikacji kodu programu w oparciu o zdefiniowany zbiór reguł i kryteriów, powstałych na podstawie doświadczenia, standardów, wcześniejszych obserwacji i przemyśleń Listy kontrolne stosuje się także w testach statycznych (np. w celu sprawdzenia zgodności interfejsu użytkownika ze standardami) Colombi - nglycza immitrija Projekt n PONL 04.01 0.10 0.02 21 10 National Guipoliki (sparij na wietry suportina i sparije na	
	1
Testowanie oparte na doświadczeniu	
☐ Testowanie eksploracyjne jest nieformalną techniką	
projektowania testów, polegającą na równoczesnym poznawaniu zachowania systemu, projektowaniu testów,	
wykonaniu testów oraz wykorzystaniu informacji zdobytej podczas testowania do zaprojektowania lepszych testów	
☐ Testowanie oparte na przeprowadzaniu ataków jest	
działaniem mającym na celu ocenę jakości, a w szczególności niezawodności programu, poprzez wymuszanie wystąpienia	
określonych awarii	
Zasada tego testowania polega na badaniu interakcji między testowanym programem, a środowiskiem, w którym program	
został uruchomiony	
Colomic - regispace investigation Projekt on POKL 04 0.10 to 02 410 9 Lectation dia gaspostavi opartin na wiedzy' wopdificansowany przez Unię Europejską w ramach środkie Europejskapi Fundazcu Gydecznego KAPITAL LUDZXI	
MATERIAY DYDACTYCZNE I SZKOLENOWE NA STLIDIA DODYDLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBLOWANE SA BEZPLATNIE 80	
	1
Testowanie oparte na doświadczeniu	
☐ Do technik testowania wykorzystujących doświadczenie	
programistów można dodatkowo zaliczyć:	
 nieformalne testowanie kodu inspekcje kodu 	
kontrole koleżeńskie	
☐ Nieformalne testowanie kodu (testowanie <i>ad-hoc</i>) jest przeprowadzane przez programistę w trakcie tworzenia	
programu (modułu), zwykle w środowisku wytwórczym	
Testowanie to ma zwykle charakter doraźny, jego najbardziej charakterystyczną cechą jest brak formalnego przygotowania	
testu (w tym zaprojektowania przypadków testowych)	
Cidoxiek - najlepaza Inwestycja Projek nr POKL 04.01.01.00.2130 / Katalacenie dla apopodarki opartej na wiedzy' współlnastoswany przez tuk Europsjak w amante ródzinie Europsjakopikungok Funduszu Społecznago	
KAPITAL LUDZKI ANDONIN TRATCIS PRINCICI FANCE PRICCE	







Testowanie oparte na doświadczeniu	
□ Inspekcje kodu to techniki statycznej analizy (formalnego przeglądu) kodu, wykonywane zgodnie z ustaloną procedurą przez powołane do tego zespoły, w celu poszukiwania najbardziej typowych błędów (usterek) w kodzie, oceny zgodności kodu z obowiązującymi standardami, wykrycia istniejących błędów, weryfikacji zastosowanych rozwiązań programistycznych, projektowania przypadków testowych Zespół przeprowadzający inspekcję kodu powinien składać się z co najmniej czterech osób (w tym autora kodu), pełniących role: prowadzącego spotkanie (moderatora), odczytującego fragmenty kodu, inspektorów szukających i identyfikujących błędy oraz protokolanta **Colombia** - Polyck Ost. 10 1-00 2-10 10 Austendendeli opadie in a wiedzy** **Prijakt UDZKI** **APTIAKL UDZKI** **MATTRIBAY O'DAKTYCZNE I SZYOL DIACOWE MA STUDIA POGOTO MOMOWE I SPECJULISTYCZNE BUISCI I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **APTIAKLUDZKI** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **SZYOLENIA** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **SZYOLENIA** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **SZYOLENIA** **SZYOLENIA** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **SZYOLENIA** **SZYOLENIA** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **SZYOLENIA** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **SZYOLENIA** **SZYOLENIA** **APTIAKLUDZKI** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA DYSTRYBLIOMANE SA BEZPAATNE** **SZYOLENIA** **SZYOLENIA** **SZYOLENIA** **MATTRIBAY O'DAKTYCZNE I SZYOLENIA** **SZYOLENIA** **S	
	1
Testowanie oparte na doświadczeniu	
Proces inspekcji kodu można podzielić na kilka etapów: planowanie procesu inspekcji (wybór członków zespołu, logistyka) indywidualne przygotowywanie się do spotkań cykliczne spotkania członków zespołu, w trakcie których:	
o poszukuje się błędów o przekazuje się ich listę autorowi o weryfikuje się poprawki w kodzie i ewentualnie podejmuje się	
decyzję o powtórnej inspekcji Inspekcje kodu są bardzo efektywną techniką znajdowania błędów, która pozwala średnio wykryć od 60% do 90% wszystkich błędów w kodzie, a tym samym znacząco poprawić jego jakość	
Colosiels - regilippeza investrigia Projek ve PORC 5.0 01 o regilippeza investrigia Projek ve PORC 5.0 01 o regilippeza investrigia APITAL LUDZKI AND TRACE VENDEN AND TR	
	1
Testowanie oparte na doświadczeniu	
☐ Kontrole koleżeńskie są prostą formą przeglądu kodu, realizowaną najczęściej z udziałem programisty – autora kodu i jednego lub dwóch programistów z tego samego zespołu, których celem jest znalezienie błędu lub weryfikacja	
zastosowanych rozwiązań programistycznych Zarówno podczas inspekcji kodu jak i w trakcie kontroli koleżeńskich wykorzystuje się:	
 listy najczęściej popełnianych błędów (uzupełniane po przeglądzie) narzędzia wspomagające statyczną analizę kodu, których zadaniem jest wyszukiwanie potencjalnych błędów (niezainicjowane zmienne, niewykorzystane zmienne, możliwe przekroczenia zakresu tablic, 	
niezgodności typów, niewywołane funkcje, niewykorzystane wyniki przypisań, nieosiągalny kod, funkcje potencjalnie niebezpieczne,) **Cabanie** - replicaci intensipcja **Projekt nr POKL 54 0.1 0.1 0.0 0.2 110 1 Katalina da ganodoski opastej na wiedzy ***sepúdlinarowany przez thnę Europijsky a manich dostok Europijsky Pundaza Gyodocznego ****CAPITAL LUDZXI	



KAPITAL LUDZKI





Tostovranja aparta na doćurjadozanju	
Testowanie oparte na doświadczeniu	
☐ Podsumowanie:	
 wiedza i doświadczenie testera jest najczęściej czynnikiem, który 	
zwiększa skuteczność wykrywania błędów testowanie oparte na doświadczeniu obejmuje zarówno działania	
zaplanowane jak i doraźne, będące reakcją na zdarzenia, mające	
miejsce podczas przeprowadzania wcześniejszych testów tego typu testowanie jest przeważnie użyteczne wtedy, gdy brakuje	
czasu na zaprojektowanie i wdrożenie procedur testowych lub kiedy	
program jest słabo udokumentowany testy wykorzystujące doświadczenie testera bardzo czesto stanowia	
istotne uzupełnienie testów strukturalnych i funkcjonalnych,	
eliminując niektóre z ich słabości	
Caloniek – najlepsza inwestycja	
Projekt nr POKL 04.01.01-00-21310 (Astralacene dia geagodaria logante) na wiedzy współlinansowany przez Unię Europejskią w ramani środkiw Europejskiągo Funduszu Społecznego (APITAL LUDZKI)	
MATERIAN DYDAKTYCZNE I SZKOLENIOWE NA STLIDIA PODVEJ OMOVJEL SPĘCIALISTYCZNE KURSY I SZKOLENIA DYSTRYBLIOWANE SA BEZPANTNIE	
Planowanie i dokumentowanie testowania	
	-
☐ Podstawowym standardem dotyczącym dokumentowania testowania oprogramowania jest IEEE 829, który specyfikuje	
formę dokumentów produkowanych i wykorzystywanych	
w różnych fazach testowania	
☐ Dokumenty związane z przebiegiem testowania:	-
 plan testów (test plan) specyfikacja warunków testowania (test design specification) 	
 specyfikacja war uników testowania (test design specification) specyfikacja przypadków testowych (test case specification) 	-
 specyfikacja procedur testowych (test procedure specification) 	
 raporty i dzienniki dotyczące użycia przypadków testowych (test item transmittal report, test log) 	
 raporty wyników testowania (test incident report, test summary 	
Projekt or POKI. 04.10-10-2130 / Scatalock – najlepsza inventycjia Projekt or POKI. 04.01-00-21310 / Scatalocenie dla gospodanki opartej na wiedzy współlinanzowany przez biel puspojekja v ramach decidno Europejskiogo Funduszu Społecznego	
współinansowany przez Unię Europejską w ramach środków Europejskiego Funduszu Społecznego KAPITAŁ LUDZKI KAPONE POKICKO WINKELE WYMOŚĆ	
MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STILDIA PODYPLOMOWE I SPĘCIALISTYCZNE KLIRSY I SZKOLENIA DYSTRYBLIOWANE SA BEZPŁATNIE 86	
	1
Planowanie i dokumentowanie testowania	
☐ Plan testów jest jedno- lub wielopoziomowym dokumentem	
opisującym planowanie procesu testowania	
Plan testów zawiera m.in. następujące informacje:	
 lista członków zespołu testującego przedmiot i zakres testowania 	
 znane ograniczenia procesu testowania 	
strategie/metody testowania	
 opis środowiska testowego/zasobów niezbędnych do testowania lista zadań testowych 	
harmonogram czynności testowych	
 monitorowanie i nadzór nad przebiegiem testowania 	
 kryterium zakończenia testów Człowiek - najpepaza inwestycja 	

Człowiek – najlepsza inwestycja







Planowanie i dokumentowanie testowania	
☐ Specyfikacja warunków testowania jest dokumentem opisującym w sposób szczegółowy czynniki lub zdarzenia, które mogą być weryfikowane (pokrywane) przez jeden lub więcej przypadków testowych, identyfikującym powiązane ze sobą przypadki testowe wysokiego poziomu oraz definiującym kryteria przejścia testów	
□ Specyfikacja przypadków testowych jest dokumentem zawierającym zbiór wszystkich przypadków testowych, zaprojektowanych z uwzględnieniem specyfikacji warunków testowania i ułożonych w porządku ułatwiającym ich wykonanie, zgodnie z przyjętą procedurą testową	
Projekt or PORL, 0.4.0.1.0.1.00.2130 / Scattelanne dia grappostari spartigi na wiedzy wgodfinanzionary przez Linię Europsjąką w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Europsjąka w ramach fotodow Europsjąksop Funduszu Spidecznego wgodfinanzionary przez Linię Funduszu Spidecznego wgodfin	
Planowanie i dokumentowanie testowania	
☐ Każdy zaprojektowany przypadek testowy powinien zostać udokumentowany przez podanie: • unikalnej nazwy (id)	
 autora opisu zastosowania warunków początkowych, jakie powinny zostać spełnione przed rozpoczęciem testowania warunków początkowych i końcowych innych obiektów wchodzących 	
w skład IUT danych wejściowych / czynności (lub sekwencji czynności) do wykonania coczekiwanego wyniku końcowego (zwracane wartości, rzucane wyjątki, generowane komunikaty, stan wyjściowy IUT) Calculat - najpora i irozki, Calculat - najpora i irozkojo popia r poku, Calculat - najpora i irozkojo popia ry poku, Calculat - najpora i irozkojo popia ry poku, Calculat - najpora i irozkojo i irozki popia ry poku popia y aranta fodora ikunopalejog a irozka i	
wapdifinandowany przez Unię Europojską w ramach środow Europojskiego Funduszu Społecznego CAPITAL LUDZKI ANDERSON POWIE POWI	
Planowanie i dokumentowanie testowania	
☐ Specyfikacja procedur testowych jest dokumentem opisującym sposób przeprowadzania każdego testu, w tym: założenia testu oraz sekwencje kroków testowania	
W przypadku automatycznego wykonywania testu, sekwencje podejmowanych akcji zostają opisane w skrypcie testowym, który jest zautomatyzowaną procedurą testową	
☐ Procedury testowe mogą mieć przydzielone priorytety, które decydują o kolejności przeprowadzania testów Wykonywanie w pierwszej kolejności najważniejszych	
procedur testowych zwykle pozwala na osiągnięcie przyjętych celów testowania w najbardziej efektywny sposób	
Cotowide - najlepoza investrycju Projekt nr POKL 04.010 / 100 / 1210 / N. Estalenia dia oppoziahri oparteji na wiedzy współlinansowany przez Uhije Europojską w zamach środów Europojskogo Funduzau Spółecznego wsa. REMONIAN	

Człowiek – najlepsza inwestycja 4 01 01-00-213/10 - Kształcenie dla gospodarki d





Planowanie i dokumentowanie testowania	
□ Raporty i dzienniki dotyczące użycia przypadków testowych opisują elementy testowe (ich konfiguracje, status i inne informacje przekazywane na początku fazy testowania od programistów) oraz chronologiczny zapis szczegółów związanych z wykonanymi testami □ Raporty wyników testowania zawierają listę zgłoszonych błędów, ocenę testowanych elementów pod względem zgodności z przyjętymi kryteriami, różne statystyki podsumowujące aktywności testowe i osiągnięte wyniki testowania, ocenę postępu procesu testowego w stosunku do założeń z planu testów, wyciągnięte wnioski dotyczące oceny ryzyka projektu oraz zalecenia dla kadry menadżerskiej Chowie- najkapia imensycy Chowie- najkapia imensycy Chowie- planymi mensycy Chowie- planymi mensycy	
Dianamania i dalamantamania tantamania	1
Planowanie i dokumentowanie testowania Na etapie ustalania strategii testowania zakłada się najczęściej podział procesu testowania na kilka odrębnych faz, zależnych od zastosowanego modelu wytwarzania oprogramowania W trakcie planowania testowania należy jasno sprecyzować, jakie kryteria muszą być spełnione, żeby zakończyć jedną fazę testowania i przejść do kolejnej fazy	
□ Typowe fazy (poziomy) testowania oprogramowania: • testowanie jednostkowe • testowanie integracyjne • testowanie systemowe • testowanie akceptacyjne • testowanie akceptacyjne • pojek ny POKL 0.6 0.10 10 0.00 2310 "N. Katikaenie dla geopodani opanie na wiedzy" **Pojek ny POKL 0.6 0.10 10 0.00 2310 "N. Katikaenie dla geopodani opanie na wiedzy" ***POJEK NY DO 2310 "N. Katikaenie dla geopodani opanie na wiedzy" ****POJEK NY DO 2310 "N. Katikaenie dla geopodani opanie na wiedzy" ***********************************	
	1
Testowanie jednostkowe	
☐ Testowanie jednostkowe (testowanie modułów) jest metodą testowania strukturalnego, której celem jest sprawdzenie poprawności działania pojedynczych i nietrywialnych jednostek kodu (elementów składowych oprogramowania), takich jak funkcje, procedury, klasy, metody (zwłaszcza	
takich jak runkcje, procedury, klasy, metody (zwiaszcza publiczne) lub zbiory współpracujących ze sobą klas ☐ Testowana jednostka kodu jest zazwyczaj wykonywana w środowisku, w którym wytwarzane jest oprogramowanie, a jej zaobserwowane zachowanie (zwracany wynik, stan	
obiektu, generowane komunikaty, rzucony wyjątek) jest porównywane z zachowaniem oczekiwanym Colonele - najdopsza irwedycja Popiek nr POKL 0.6 do 10 40 21 10 11 Kartikenie do popodań spartej na wiedzy spedifinancowany przez tile fezpojek a nramat bodnie Europspiekop Funduszu Spelezznego	
KAPITAL LUDZKI	I







_ Testowanie jednostkowe	
 Cechy testowania jednostkowego: testowanie jednostek kodu (względnie małych fragmentów kodu) jest na ogół łatwiejsze i efektywniejsze od testowania całego programu testowanie poszczególnych jednostek nie wymaga uruchamiania całego programu, a jedynie jego niezbędnej do testowania 	
i wyizolowanej z całości części kodu testowanie jednostkowe można łatwo zrównoleglić testowanie jednostkowe bardzo łatwo zautomatyzować testowanie jednostkowe może dotyczyć:	
wyłącznie metod publicznych wszystkich metod o nietrywialnym zachowaniu testowanie jednostkowe nie jest równoważne testowaniu kompletnego, modularnego programu Człowienajpoga innegłycja Człowienajpoga innegłycja	
Colonies - Analysisca investiga in meeting in Projekt ner POKL, Out 0.10 to 0.02 1310 (Australend and grappdank opastig na wedzh)* Projekt ne POKL, Out 0.10 to 0.02 1310 (Australend and grappdank opastig na wedzh)* (APITAL LUDZKI MATERIAN FORMATONING SERVI ENIONE IN SERVICE AND SERVI ENION SERVICE AND	
Testowanie jednostkowe	
□ Dobrze zaprojektowane testy jednostkowe powinny być: • niezależne – jeden test nie może mieć wpływu na pozostałe testy • powtarzalne – wielokrotne wykonanie testu dla danego zestawu danych powinno dawać taki sam wynik • jednoznaczne – testy powinny dawać jasną odpowiedź dotyczącą testowanej funkcjonalności	
 jednostkowe – testujące jedną funkcjonalność na raz kompletne – testujące wszystko co może zawieść hermetyczne – nieodwołujące się do zewnętrznych zasobów 	
☐ Zbiór przypadków testowych, który dotyczy jednej jednostki kodu grupuje się zazwyczaj wewnątrz jednej dedykowanej klasy, zwanej klasą testującą Control - nającaza inwestycja (na wietry) Projekt n POLI, d. d. d. 17 - nającaza inwestycja (na wietry) **repolitation (na d. d. 17 - nającaza inwestycja) **repolitation (na d. 17	
CAPITAL LUDZKI WASSESS MALES PROMOTE STAND INFORM NA STUDIA POONS ORANGE SISSE AUSTRALISES STAND A TOTAL PROMOTE STAND INFORM NA STUDIA POONS ORANGE SISSE AUSTRALISES STAND A TOTAL PROMOTE STAND INFORM NA STUDIA POONS ORANGE SISSE AUSTRALISES SISSOLITINA TOTAL POONS ORANGE SISSE SISS	
	1
Testowanie jednostkowe	
☐ W przypadku testowania obiektów, które są bardzo silnie ze sobą powiązane, działają powolnie lub niedeterministycznie, są trudne do skonfigurowania, związane z interfejsem	
użytkownika lub po prostu jeszcze nie istnieją, prawidłowe przeprowadzenie testowania jednostkowego może być znacząco utrudnione	
W takich przypadkach rzeczywiste obiekty są na czas uruchamiania i testowania kodu zastępowane przez obiekty "imitacji" (<i>mockobjects</i>), posiadające takie same interfejsy	
jak obiekty rzeczywiste i dostarczające testowanym obiektom wcześniej przygotowane do testów dane	
Calosiels — najleptaz investycja: Projele nr POKL 04.01.01-00:21910 Nctationeni dia grappotanti opantiji na wiedný* współlinansowany przez Linię Europojski, w ramach rodorów Europojskiego Funduzzu Społecznego (APITAL UDZX) www.eatonsow.	





Testowanie jednostkowe					
☐ Przykład: Testowanie jednostkowe metody lastOddIndex(), wykorzystujące narzędzie jUnit dla języka Java					
<pre>public class ArrayTools { public static int lastOddIndex(int[] table) { int index = -1; for (int i = 0; i < table.lenght-1; i++) { if (tabli] % 2 == 0) { index = i; } } return index; } }</pre>					
Cdowek - nglippzas investrycja Projekt nr PCRL 54.01.01-00 22.13.01.4.Catchevini dla grappolarki opartini na viestry' sepolifinareneswy przez thet Europskik v w Worksche Europskik pri midsan Glodecanego CAPITAL LUDZXI					

Testowanie jednostkowe					
<pre>import org.junit.*; public class TestArrayTools { @Test public void testLastOddindex() { assertEquals(-1, ArrayTools.lastOddindex(new int[] {2, 0, 8})); assertEquals(-1, ArrayTools.lastOddindex(new int[] {0})); assertEquals(2, ArrayTools.lastOddindex(new int[] {1, 2, 7})); assertEquals(2, ArrayTools.lastOddindex(new int[] {2, 21, 5})); assertEquals(2, ArrayTools.lastOddindex(new int[] {2, 2, 6})); assertEquals(0, ArrayTools.lastOddindex(new int[] {1, 2, 6})); </pre>					
Projekt nr POKL 04.01.01.00.021910, Actalorene dia propoderia oparteji na wiedzy wopdifinansowany przez Unie Europejską w ramoch środkie Europejskogo Funduszu Spidecznego APITAL UDZXI ANIOSAN FULIO WIEGOWANIA SPIDINA POTYPO (MONOYE 1987 CILII STYLYZINE KURSY) SYKOLENIA INVICANIA WARIONA SPIZINA POTYPO (MONOYE 1987 CILII STYLYZINE KURSY) SYKOLENIA INVICANIA SIGERPANTINE MATERIAŁY DYMAKTYZNE I SYKOLENIA NA STILINIA POTYPO (MONOYE 1987 CILII STYLYZINE KURSY) SYKOLENIA INVICANIA SIGERPANTINE 95					

☐ Testowanie różnicowe jest metodą określania efektywności testu tj. skuteczności w znajdywaniu błędów w kodzie☐ Ideą tego podejścia jest tworzenie odmian (mutantów) testowanego kodu poprzez dokonywanie w nim niewielkic modyfikacji Różne mutanty kodu podlegają następnie testowaniu, a wszelkie zauważone rozbieżności w wynikach testowania sa rejestrowane i analizowane	Testowanie różnicowe (mutacyjne)			
Różne mutanty kodu podlegają następnie testowaniu, a wszelkie zauważone rozbieżności w wynikach testowania	ów w kodzie ı (mutantów)			
są rejestrowane rananzowane				
☐ Dobrze zaprojektowany test powinien "zauważyć" zmianę w zachowaniu programu, wynikającą z wprowadzonej modyfikacji w kodzie	, ,			
Cáseleia - najlecca investirója Projekt or POXLO 6/10 1/0 0/2110/ Szetalende da grapodoski spateji na viedzy' vegoffinansowany przez Unię Europejską w ramach środkow Europejskogo Funduszu Sp-decznego www.samon.or. racyczonosowany racyczonosowany racyczonosowany racyczonosowany racyczonosowany racyczonosowany racyczonosowany racyczonosowany racyczonosowany				







Testowanie różnicowe (mutac	:yjne)	
☐ Przykład:		
// kod oryginalny double sum(double a, double b) { return a + b; } // mutant #1 double sum(double a, double b) { return a * b; } // mutant #2 double sum(double a, double b) {	Oceń jaka jest efektywność poniższych testów: TC1: {a = 0, b = 0} TC2: {a = 1, b = 1} TC3: {a = 1, b = 0} TC4: {a = 2, b = 2} TC1 nie wykrywa mutantów, TC2 wykrywa oba mutanty,	
return a + 0; }	TC3 wykrywa mutant #1, TC4 wykrywa mutant #2	
Projekt nr POKL 04.01.01.00.2130 (Accident - najepaza inneval Projekt nr POKL 04.01.01.00.2130 (Accident - najepaza inneval repolitionarcowany przez Unię Europejską w ramach środnow św. KAPITAŁ UDZKI MATERIAŁY KYNATYCYNE I STWYL ENIYME MA STUDNA PONYOR MAMOEL I SECYLALIST	roja copodania opantej na wiedzy* suropejskiego Funduszu Społecznego www.sawinusw. TVYZNE V I I SCY / SZYCHEMIA DYCTOVO I I MANANE SA BEZDA ATMIE 100	
MAIL-HIARY DYDAK IYEZNE I SZKOLENIOWE NA STUDIA PODVPLOMOWE I SPECIALIST	TLENE ALJISY I SZKULENIA DYSTRYBUOWANE SA BEZPŁATNIE 100	
		_
Testowanie integracyjne		
	1 .	
 Celem testowania integracyjneg współpracy pomiędzy modułam komponentami) oprogramowan podzespoły 	i (jednostkami,	
☐ Testy integracyjne są wykonywa	ine przez programistów	
odpowiedzialnych za proces sca w środowisku, w którym wytwa		
i służą głównie wykrywaniu błę	dów w interfejsach oraz	
interakcjach pomiędzy łączonyn ☐ Proces scalania i testowania pos		
wielokrotnie powtarzany, aż do programu	zbudowania kompletnego	
<i>Edowiek – najlepsza inwest</i> Projekt nr POKL 04.01.01-00-213/10 /Ksztalcenie dia swspółfinansowany przez Unię Europejsky w ramach środków E	roja gospodarki opartej na wiedzy" Europejskiego Funduszu Społecznego	
KAPITAL LUDZKI ANDORNI PRATICI SPRANSCI MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODYPLOMOWE I SPECIALIST	UNIA EUROPERA E ADDRES POLICEN TYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA BEZPŁATNIE 101	
Testowanie integracyjne		
☐ Testy integracyjne opracowuje s programu, najczęściej jeszcze pr	ię na podstawie specyfikacji zed fazą kodowania	
☐ Testowanie integracyjne należy powstaniu zdatnych do użycia m	rozpocząć natychmiast po	
sposób jego przeprowadzania zależy od przyjętej strategii scalania modułów		
☐ Strategie testów integracyjnych:		
przyrostowazstępująca (top-down)		
o wstępująca (<i>bottom-up</i>) ■ skokowa (<i>big bang</i>)		
Criminis - milinores insuents	arrin	
Projekt nr POKL-04.01.01-00 23/310 (X-Stationini dila współfinansowany przez Unię Europejską w ramach środków E KAPITAŁ LUDZKI MACKOWA POKACA SPOKACA SPOKACA SPOKACA SPOKACA SPOKACA SPOKACA SPOKACA SPOKACA SPOKACA SPOKAC	ycja gospodarki opartej na wiedzy" Luropejskiego Funduszu Społecznego	

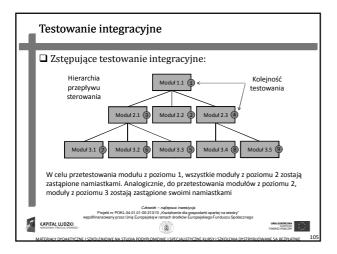






Testowanie integracyjne	
☐ W podejściu przyrostowym moduły lub systemy są scalane i testowane stopniowo, jeden po drugim (lub po kilka na raz), dopóki wszystkie komponenty programu nie zostaną ze sobą zintegrowane i przetestowane	
☐ W podejściu skokowym testowanie modułów następuje dopiero po ich scaleniu w duże podzespoły lub w kompletny program	
Podstawową wadą podejścia skokowego jest to, że błędy zostają wykryte w bardzo późnej fazie procesu testowania, co utrudnia ich zlokalizowanie oraz zwieksza koszty	
i podnosi ryzyko projektu	
Projekt er POXLO 4.0 II.01.00.213/01, Asstatemen dia geopodetri opatrij na wiedzy wopodifinancewany przez Unię Europejską w ramach brodidwe Europejskago Funduszu Społecznego (APITAL LUDZKI MATERIAN DYDAKTYCZNE I SZNOLENIOWE NA STUDIA PODPYLOMOWE I SPECIALISTYCZNE KURSY I SZNOLENIA DYSTRUBLOWANE SA BEZPEJZNE 103	

Testowanie in	egracyjne			
i testuje się m	wania metodą zstępujo duły programu będą duły na niższych pozi	ce na szczycie h	ierarchii,	
namiastkami	<i>stubs</i>), tj. modułami o ecz o ograniczonej fun	takich samych		
modułów zna przy czym kol	e moduły są następnie dujących się na niższy ejność testowania mo archii jest dowolna	m poziomie hie	erarchii,	
☐ Ten proces jest powtarzany, aż do przetestowania modułów znajdujących się na najniższym poziomie hierarchii, co				
, , , , , , , , , , , , , , , , , , ,	testowania całego pr	O		
Projekt współfinansowa KAPITAŁ LUDZKI KARJOGINIA STRATIGIA SPRINOŚCI	r POKL 04.01.01-00-213/10 .Kształcenie dla gospodari y przez Unię Europejską w ramach środków Europejski III	ı opartej na wiedzy" ago Funduszu Społecznego u Fak	INIA EUROPESSA BIOCESCO NORSZ GORCZNY	

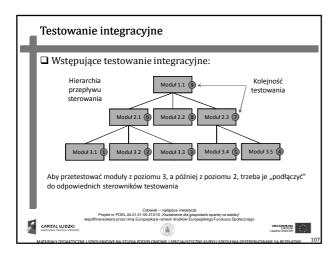








Testowanie integracyjne
☐ Podczas testowania metodą wstępującą najpierw scala i testuje się moduły na najniższym poziomie hierarchii, wykorzystując moduły-sterowniki (<i>drivers</i>) testowania do symulowania ich wywołań przez moduły położone wyżej w hierarchii
☐ Przetestowane moduły są następnie używane do testowania modułów znajdujących się na wyższym poziomie hierarchii, przy czym kolejność testowania modułów na tym samym poziomie hierarchii jest dowolna
☐ Proces testowania całego programu uznaje się za zakończony po przetestowania modułów znajdujących się na najwyższym poziomie hierarchii
Projekt nr PORL 0.4 01 0.0 219 10. Actalizanie dla genociatri spanie na wiedzy wapolificania observanje na wiedzy wapolificania observy przez Unię Europejską u Porta na observanje na wiedzy wapolificania observy przez Unię Europejską u Porta na observanje observanje na observanje n



Testowanie integracyjne				
	□ Podsur	nowanie:		
	 proje pros i reje rzec 	Novanie: ektowanie i implementowanie modułów-sterowni tsze od namiastek (sterowniki przekazują dane te estrują wyniki, namiastki muszą natomiast symulc zywistych modułów, od działania których zależą w owania)	stowe wać działanie	
		ą testowania wstępującego jest niemożność skons sji "szkieletowej" programu (brak głównego modu		
	przy	ą testowania zstępującego jest duża trudność proj padków testowych, które muszą uwzględniać wsz acje (zmieniające się moduły pośredniczące)		
	 zaletą testowania wstępującego jest to, że można do niego przystąpić już w bardzo wczesnej fazie tworzenia programu (nie jest konieczne 			
	ukoŕ	ńczenie projektu architektonicznego) Człowiek – najlepsza inwestycja Projekt nr POKL 04.01.010-02 12/101. Kastaleonie dia ozeoodarki opartej na wiedzy*		
*	KAPITAL LUDZKI NARODOMA STRATEGIA SPOINOSCI	współfinansowany przez Unię Europejską w ramach środków Europejskiego Funduszu Społecznego	UNIA EUROPLISAA BARDYLISH FANDESZ POLICZNY	







Testowanie systemowe	
☐ Testowanie systemowe jest kolejnym poziomem testowania, w którym sprawdza się, czy funkcjonalność i jakość w pełni zintegrowanego programu jest zgodna z oczekiwaniami	
☐ Testy systemowe przeprowadza się najczęściej w oparciu o specyfikację wymagań dla programu (uwzględniającą zarówno wymagania funkcjonalne, jak i niefunkcjonalne) lub różne wysoko-poziomowe modele opisujące jego funkcjonowanie (scenariusze użycia, procesy biznesowe, modele transakcji)	
☐ Testy systemowe są przeprowadzane przez programistów lub niezależny zespół testowy, najlepiej w środowisku maksymalnie zbliżonym do docelowego Cotowei - najproza imentoja Cotowei - najproza imentoja Projek nr POKL 6.0.0 10 00 210 10 Kataltoma da gospodanią oparteja na wiedzy' spopliżanowane przez bie leśnickie Europejskieja rhamkaru Społecznego	
MARTHER LIDEX I SECOND FOR THE PROPERTY OF THE	
Testowanie systemowe	
☐ Testowanie systemowe może uwzględniać przydzielone poszczególnym funkcjom priorytety po to, aby w pierwszej kolejności były testowane najważniejsze dla klienta funkcje programu	
□ Przegląd wybranych testów systemowych sprawdzających różne aspekty funkcjonowania programu: ■ Testy funkcjonalne służą do sprawdzania działania programu pod kątem jego zgodności ze specyfikacją (strategia testowania "czarnej skrzynki") ■ Testy wydajnościowe służą do pomiaru i oceny czasu realizacji	
wybranych operacji (szybkości przetwarzania danych, liczby realizowanych w jednostce czasu transakcji, szybkości odczytu/ zapisu, czasu reakcji na zdarzenia,) Cachaela najpasu insweptoja w czesk odcine insweptoja w cycholicznego podnie zapostoja w podlicznowany przez liche podpoja w ranche fodnie Eucopspiejacje frundacia Społecznego	
CAPITAL LUDZXI WARDONE HINTON PRINCIP	
	1
Testowanie systemowe	
 Testy obciążeniowe służy do pomiaru i oceny zachowania programu przy zwiększającym się obciążeniu (liczbie równolegle pracujących użytkowników, liczbie transakcji,), oraz do sprawdzenia, przy jak dużym obciążeniu program jest jeszcze w stanie prawidłowo działać Testy przeciażające służa do oceny zachowania programu na granicy 	
 lesty przeciązające srużą do oceny zacnowania programu na granicy lub poza granicami wyspecyfikowanych wymagań Testy interfejsu użytkownika służą głównie do oceny jego poprawności funkcjonalnej oraz użyteczności (jakości i solidności wykonania, ergonomii, stopnia trudności jego obsługi, łatwości 	
uczenia się interfejsu, szybkości działania, odporności na błędy użytkownika, zdolności do adaptacji,) Testy dostępności służą do oceny w jakim stopniu zastosowane rozwiązania pozwolą na użytkowanie programu przez osoby	
z dysfunkcjami (niewidome, niedowidzące, z zaburzoną motoryką) Człowieł - rajdpzza inweniycja Projekt ny DNL. D.4. D.4. O.2 23 10 1. Katalenienie da podnicki oparteja na wiedzy' wepdifizacionawa y zerze Unie projekty na zerzely odobnie Europejskiego produca Spieloznego ANTAL UDZIA	







Testy akceptacyjne	
□ Testowanie akceptacyjne jest finalnym etapem testowania zaimplementowanego programu, mającym na celu sprawdzenie, czy spełnia on wyspecyfikowane oczekiwania użytkownika i realizuje założone procesy biznesowe □ Testy akceptacyjne przeprowadzane są przez użytkowników systemu lub ich reprezentantów przy współudziale przedstawicieli producenta, w docelowym środowisku pracy (z uwzględnieniem sprzętu i systemu operacyjnego) □ Testy akceptacyjne pozwalają formalnie ocenić jakość stworzonego oprogramowania, a także zweryfikować niejawne założenia i oczekiwania, poczynione zarówno przez klienta, jak i zespół producenta oprogramowania Comunic najwa manaby przez klienta, jak i zespół producenta oprogramowania Comunic najwa manaby przez klienta, jak i zespół producenta oprogramowania Comunic najwa manaby przez klienta, jak i zespół producenta oprogramowania Comunic najwa manaby przez klienta, jak i zespółu producenta oprogramowania CAPITAL LUDZXI. ***MATERIALY DIOACTICZNE I SZOLENIOWE NA STUDIA PODPROMOWI I SPECIALISTYCZNE KURCY I SZOLENIO POSTATRIBLOWANI SA BEZPATNE** ****TERIAL NIDZXI.** *********************************	
	_
_ Testy akceptacyjne	
☐ Testy alfa są wewnętrznymi testami akceptacyjnymi, które sa przeprowadzane przez potencjalnych użytkowników	
lub niezależny zespół testowy i odbywają się u producenta,	
ale bez udziału wytwórcy oprogramowania Testy beta są zewnętrznymi testami akceptacyjnymi, które	
są przeprowadzane przez większe grupy użytkowników	
i odbywają się poza miejscem wytwarzania oprogramowania Testowanie beta jest podejściem często wykorzystywanym	
po to, aby uzyskać zwrotną informację z rynku dla tzw.	
oprogramowania "z półki"	
Projekt nr POKL.04.01.01-00:21310 .Katalsonne dia pospodarini opantej na wiedzy'	
Pengel nr PVML Do. (0.10-00-0213 O. Katalicene da a peopodario partin ja wedaty vegoffinansowany przez Unit Europejska w mank drodkie Europejska princiszu Społecznego www.sepogram. APPTIAL LUDZKI washoos WALCO SPORT OF THE EUROPESKA W MANKET OF THE EUROPESKA SPORT OF THE E	
MATERIAN DYDAKTYCZNE I SZKOLENIOWE NA STLIDIA PODVPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBLIOWANE SA BEZPLATNIE 113	
	1
Niezawodność oprogramowania	
☐ Niezawodność, obok takich cech jak: funkcjonalność, użyteczność, wydajność i wiele innych, jest elementem	
istotnie wpływającym na jakość oprogramowania	
☐ Niezawodność oprogramowania (wyrażana w sposób jakościowy lub ilościowy, jako wartość pewnej miary) jest	
często kryterium pojawiającym się w wymaganiach klienta	
□ Określenie poziomu niezawodności pozwala: • oszacować koszty serwisu, liczbę personelu, nakłady środków na	
konserwowanie oprogramowania	
 ocenić i wdrożyć lepsze procedury wytwarzania oprogramowania, pozwalające na minimalizację ryzyka związanego z wytwarzaniem 	
oprogramowania, redukcję kosztów, zwiększenie reputacji firmy	
Colombin - Indigenzar inventingia Projek nr POKIL 04.01.0 to 20-1310 Australiand and paraposahri opantin na winder/ wepdifinansowany przez Dinę Europejską w ramach śrotkine Europejskago Funduszu Społecznego KAPITAL LUDZXI WAN-MARKANA	







Niezawodność oprogramowania	
☐ Miary niezawodności oprogramowania:	
 częstotliwość występowania błędnych wykonań – dla systemów 	
nietransakcyjnych, liczba błędnych wykonań przypadająca na jednostkę czasu	
 prawdopodobieństwo błędnego wykonania – dla systemów 	
transakcyjnych, częstość występowania nieudanych (na skutek błędu) transakcji	
 średni czas pomiędzy błędnymi wykonaniami 	
 dostępność – procentowo wyrażony stopień w jakim system będzie dostępny dla użytkownika wtedy, gdy będzie wymagane jego użycie (miare te można oszacować na podstawie stosunku czasu, w którym 	
system działa prawidłowo, do czasu potrzebnego na naprawę błędów, skutkujących niedostępnością systemu)	
Całowiek – najłopsza inwestycja Projekt nr POKL 04.01.01-00-213/10 "Ksztalcenie dla gospodarki opartej na wiedzy"	
współfinandowany przez Unię Europejską w ramach źrodków Europejskiego Funduszu Społecznego KAPITAL LUDZKI LIDY STANDARDOW STANDARD	
MATERIAŁY DYDAKTYCZNE I SZKOLENIOWE NA STUDIA PODVPLOMOWE I SPECIALISTYCZNE KURSY I SZKOLENIA DYSTRYBUOWANE SA BEZPŁATNIE 115	
Niezawodność oprogramowania	
☐ Niezawodność systemu wzrasta, jeżeli w trakcie poprawy zdiagnozowanych błędów, nie wprowadza się do kodu	
nowych błędów	
W przypadku testów statystycznych wzrost niezawodności ma charakter logarytmiczny, wyrażony wzorem:	
ma charakter logarytmiczny, wyrazony wzorem: niezawodność = niezawodność_początkowa * exp(-C * liczba_testów)	
stałą C należy oszacować dla każdego systemu na podstawie obserwacji statystycznych jego niezawodności	
☐ Szybszy wzrost niezawodności systemu można osiągnąć	
poprzez odpowiedni (nielosowy) dobór danych testowych	
Człowiek – najlepsza inwestycja Projek nr POKL 04.01.01-00-21310 "Ksztalcenie dla gospodarki opartej na wiedzy" wsoślitannowanne rznez libie Furnoriski war uramski frontika Furnoriskiano Furnduszu Sooleczneno	